Common Criteria for Information Technology Security Evaluation

CCEB-96/013

Part 3: Security assurance requirements

Version 1.00

96/01/31

Foreword

Following extensive international cooperation to align the source criteria from Canada (CTCPEC), Europe (ITSEC) and the United States of America (TCSEC and Federal Criteria), version 1.0 of the *Common Criteria for Information Technology Security Evaluation* is issued for the purpose of trial evaluations and for review by the international security community. The practical experience acquired through trial evaluations and all the comments received will be used to further develop the criteria.

A template for reporting observations on version 1.0 of the CC is included at the end of this document. Any observation reports should be communicated to one or more of the following points of contact at the sponsoring organisations:

National Institute of Standards and Technology
Computer Security Division
NIST North Building, Room 426
Gaithersburg, Maryland 20899
U.S.A.
Tel: (+1)(301)975-2934, Fax:(+1)(301)926-2733
E-mail:csd@nist.gov
http://csrc.ncsl.nist.gov

National Security Agency
Attn: V2, Common Criteria Technical Advisor
Fort George G. Meade, Maryland 21122
U.S.A.
Tel: (+1)(410)859-4458, Fax:(+1)(410)684-7512
E-mail: common_criteria@radium.ncsc.mil

Communications Security Establishment
Criteria Coordinator
R2B IT Security Standards and Initiatives
P.O. Box 9703, Terminal
Ottawa, Canada K1G 3Z4
Tel:(+1)(613)991-7409, Fax:(+1)(613)991-7411
E-mail:criteria@cse.dnd.ca
ftp:ftp.cse.dnd.ca
http://www.cse.dnd.ca

UK IT Security and Certification Scheme
Senior Executive
P.O. Box 152
Cheltenham GL52 5UF
United Kingdom
Tel: (+44) 1242 235739, Fax:(+44)1242 235233
E-mail: ccv1.0@itsec.gov.uk
ftp: ftp.itsec.gov.uk
http://www.itsec.gov.uk

Bundesamt für Sicherheit in der Informationstechnik
Abteilung V
Postfach 20 03 63
D-53133 Bonn
Germany
Tel: (+49)228 9582 300, Fax:(+49)228 9582 427
E-mail:cc@bsi.de

Service Central de la Sécurité des Systèmes d'Information
Bureau Normalisation, Critères Communs
18 rue du docteur Zamenhof
92131 Issy les Moulineaux
France
Tel: (+33)(1)41463784, Fax:(+33)(1)41463701
E-mail:ssi28@calvacom.fr

Netherlands National Communications Security Agency
P.O. Box 20061
NL 2500 EB The Hague
The Netherlands
Tel: (+31).70.3485637, Fax:(+31).70.3486503
E-mail: criteria@nlncsa.minbuza.nl

This document is paginated from i to x and from 1 to 174

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Scope

1 Part 3 defines the assurance requirements of the CC. It includes the evaluation assurance levels (EALs) that define a scale for measuring assurance, the individual assurance components from which the assurance levels are composed, and the criteria for evaluation of PPs and STs.

1.2 Organisation of Part 3

2 Chapter 1 is the introduction and paradigm for Part 3.

3 Chapter 2 describes the presentation structure of the assurance classes, families, components, and evaluation assurance levels along with their relationships. It also characterises the assurance classes and families found in Chapter 5.

4 Chapter 3 provides a brief introduction to the evaluation criteria for PPs and STs. It is followed by detailed explanations of the components that are used for those evaluations.

5 Chapter 4 provides detailed definitions of the EALs.

6 Chapter 5 provides a brief introduction to the assurance classes and is followed by detailed definitions of those classes.

7 Annex A provides a summary of the dependencies between the assurance components.

8 Annex B provides a cross reference between the EALs and the assurance components.

9 Annex C provides the Common Criteria observation report guidance, example observations and example printed form.

1.3 CC assurance paradigm

10 The purpose of this section is to document the philosophy which underpins the CC approach to assurance. An understanding of this section will permit the reader to understand the rationale behind the CC assurance requirements.

1.3.1 CC philosophy

11 The CC philosophy is that the threats to security and organisational security policy commitments should be clearly articulated and the proposed security measures be demonstrably sufficient for their intended purpose.

12 Furthermore, measures should be adopted which facilitate the exposure and subsequent elimination of vulnerabilities. Should elimination be impractical, measures should be adopted which minimise the impact of the vulnerability or measures should be adopted which detect any potential exploitation.

1.3.2 Assurance approach

13 The CC philosophy is to gain and quantify assurance based upon an evaluation (active investigation) of the IT product or system which is to be trusted. The CC does not exclude, nor does it comment upon, the relative merits of other means of gaining assurance.

14 Evaluation has been the traditional means of gaining assurance and is the basis for prior evaluation criteria documents. In aligning the existing approaches, the CC adopts the same philosophy but is so structured as to allow the future introduction of alternative approaches.

15 The CC proposes a measurement of assurance based upon active investigation of the IT product by expert evaluators with increasing emphasis on scope depth and rigour.

1.3.2.1 Significance of vulnerabilities

16 It is assumed that there are threat agents that will actively seek and exploit the opportunity to make illicit gains arising out of breaches of security. Due to necessity caused by the need to process sensitive information and lack of availability of sufficiently trusted products or systems it is current practice, to hold significant assets at risk to failures of IT. It is, therefore, likely that IT security breaches could lead to significant loss.

17 IT security breaches arise through the exploitation of vulnerabilities in the application of IT within business concerns.

18 Vulnerabilities within IT products and systems should therefore be exposed and, where feasible:

a) eliminated, that is active steps should be taken to remove or neutralise all known exploitable vulnerabilities;

b) minimised, that is active steps should be taken to reduce the impact of the vulnerability to an acceptable residual level;

c) monitored, that is active steps should be taken to ensure that any attempt to exploit a residual vulnerability will be detected so that steps can be taken to limit the damage.

1.3.2.2 Cause of vulnerabilities

19 Vulnerabilities can arise through failures in:

a) requirements, that is an IT product or system may possess all the functions and features required of it and still contain vulnerabilities which render it unsuitable or ineffective with respect to security;

b) construction, that is an IT product or system does not meet its specifications and vulnerabilities have been introduced as a result of poor constructional standards or incorrect design choices;

c) operation, that is an IT product or system has been constructed correctly to a correct specification but vulnerabilities have been introduced as a result of inadequate controls upon the operation.

1.3.2.3 CC assurance

20 Assurance is an attribute of an IT product or system which permits those depending on the IT product or system to have confidence that the security functions enforce the security policy. Assurance can be attributed to an IT product or system by reference to e.g., unsubstantiated assertions, prior relevant experience, or specific experience. However, the CC provides assurance through active investigation. Active investigation is an evaluation of the actual IT product or system in order to determine its actual security properties.

1.3.2.4 Assurance through evaluation

21 Evaluation has been the traditional means of gaining assurance, and is the basis of the CC approach. Evaluation techniques can include, but are not limited to:

a) analysis and checking of process(es) and procedure(s);

b) checking that process(es) and procedure(s) are being applied;

c) analysis of the correspondence between TOE design representations;

d) analysis of the TOE design representation against the requirements;

e) verification of mathematical proofs;

f) analysis of guidance documents;

g) analysis of functional tests developed and the results provided;

h) independent functional testing;

i) analysis for vulnerabilities (including flaw hypothesis);

j) penetration testing.

1.3.3 The CC evaluation assurance scale

22 The CC philosophy assumes that greater assurance results from the application of greater evaluation effort, and that the application of evaluation effort should be such as to maximise the assurance gains. The increasing evaluation effort is based upon:

a) scope, that is additional effort is deployed in evaluating a greater proportion of the IT product or system content;

b) depth, that is additional effort is deployed on evaluating greater design and implementation detail;

c) rigour, that is the additional effort is used to apply more searching tools and techniques in order to discover less obvious flaws or decrease the probability that such flaws remain.

Chapter 2

Security assurance requirements

2.1 Structures

23 The following sections describe the constructs used in representing the assurance classes, families, components, and EALs along with the relationships among them.

24 Figure 2.1 illustrates the assurance requirements defined in Part 3 of the CC. Note that the most abstract collection of assurance requirements is referred to as a class. Each class then contains assurance families which contain assurance components which ultimately contain assurance elements.

2.1.1 Protection Profile and Security Target evaluation criteria class structure

25 The requirements for protection profile and security target evaluation are treated as assurance classes and are presented using the similar structure as that used for the other assurance classes, described below. One notable difference is the absence of a component levelling section in the associated family descriptions. The reason is quite simply that each family has only a single component and therefore no levelling has occurred.

2.1.2 Class structure

26 Figure 2.1 illustrates the assurance class structure.

2.1.2.1 Class name

27 Each assurance class is assigned a unique name. The name indicates the topics covered by the assurance class.

28 A unique short form of the assurance class name is also provided. This is the primary means for referencing the assurance class. The convention adopted is an “A” followed by two letters related to the class name.

2.1.2.2 Class introduction

29 Each assurance class has an introductory section describing the composition of the class and supporting text covering the intent of the class.

2.1.2.3 Assurance families

30 Each assurance class contains at least one assurance family. The structure of the assurance families is described in the following section.

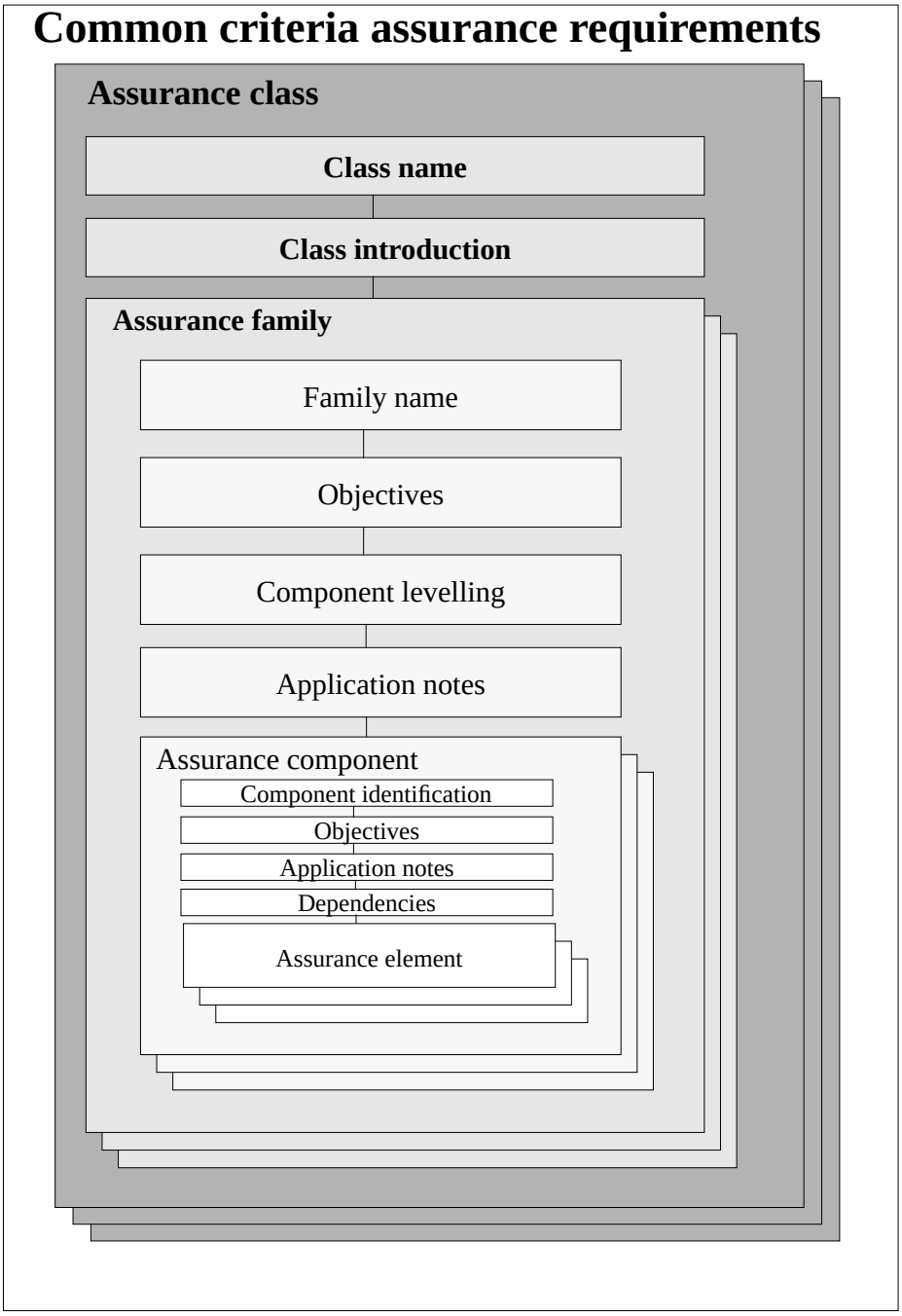

Figure 2.1 - Assurance class/family/component/element hierarchy

2.1.3 Assurance family structure

31 Figure 2.1 illustrates the assurance family structure.

2.1.3.1 Family name

32 Every assurance family is assigned a unique name. The name provides descriptive information about the topics covered by the assurance family. Each assurance family is placed within one assurance class that shares a common intent.

33 A unique short form of the assurance family name is also provided. This is the primary means used to reference the assurance family. The convention used is that the short form of the class name is used, followed by an underscore, and then three letters related to the family name.

2.1.3.2 Objectives

34 The objectives section of the assurance family presents the overall purpose and goals of the assurance family.

35 This section describes the objectives, particularly those related to the CC assurance paradigm, which the family is intended to address. The description for the assurance family is kept at a general level. Any specific details required for objectives are incorporated in the particular assurance component.

2.1.3.3 Component levelling

36 Each assurance family contains one or more assurance components. This section of the assurance family describes the components available and explains the distinctions between them. Its main purpose is to differentiate between the assurance components once it has been determined that the assurance family is a necessary or a useful part of the assurance requirements for a PP/ST.

37 Assurance families containing more than one component are levelled and rationale is provided as to how the components are levelled. This rationale is in terms of scope, depth, and rigour.

2.1.3.4 Application notes

38 The application notes section of the assurance family, if present, contains additional information for the assurance family. This information should be of particular interest to users of the assurance family (e.g., PP and ST authors, designers of TOEs, evaluators). The presentation is informal and covers, for example, warnings about limitations of use and areas where specific attention may be required.

2.1.3.5 Assurance components

39 Each assurance family has at least one assurance component. The structure of the assurance components is provided in the following section.

2.1.4 Assurance component structure

40 Figure 2.2 illustrates the assurance component structure.

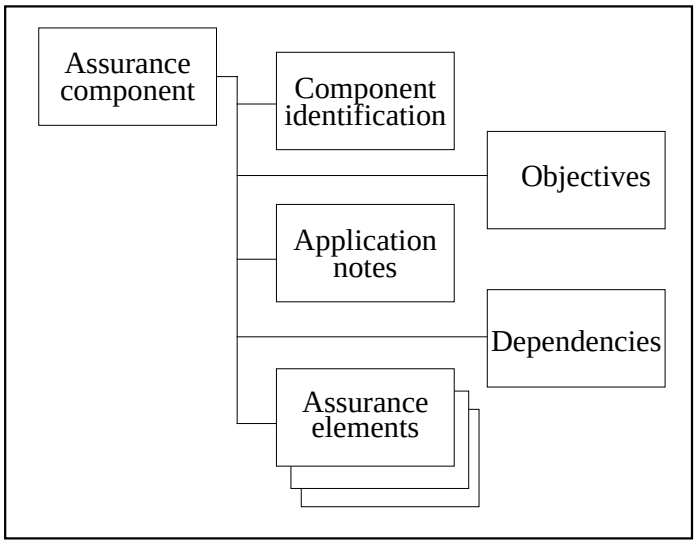

Figure 2.2 - Assurance component structure

41 The relationship between components within a family is highlighted using a bolding convention. This bolding convention includes the bolding of all new requirements. Those parts of the requirements that are enhanced or modified beyond the requirements of the previous component are bolded. In addition, any new or enhanced objectives, application notes, or dependencies beyond the previous component are also highlighted using bold type.

2.1.4.1 Component identification

42 The component identification section provides descriptive information necessary to identify, categorise, register, and reference a component.

43 Every assurance component is assigned a unique name. The name provides descriptive information about the topics covered by the assurance component. Each assurance component is placed within one assurance family that shares a common security objective.

44 A unique short form of the assurance component name is also provided. This is the primary means used to reference the assurance component. The convention used is that the short form of the family name is used, followed by a period, and then a numeric character. The numeric characters for the components within each family are assigned sequentially, starting from 1.

2.1.4.2 Objectives

45 The objectives section of the assurance component, if present, contains specific objectives for the particular assurance component. For those assurance components that have this section, it contains the specific purpose and goal of the component and a more detailed explanation of the objectives.

2.1.4.3 Application notes

46 The application notes section of an assurance component, if present, contains additional information to facilitate the use of the component.

2.1.4.4 Dependencies

47 For each assurance component, there is a complete list of the dependencies on other assurance and functional components. “No dependencies” is used to describe the situation where no dependencies have been identified.

2.1.4.5 Assurance elements

48 A set of assurance elements is provided for each assurance component. An assurance element is a security requirement which if further divided would not yield a meaningful evaluation result. It is the smallest security requirement recognised in the CC.

49 Each assurance element is identified as belonging to one of the three sets of assurance elements:

a) Developer action elements: the activities that shall be performed by the developer. This set of actions necessarily includes delivery of evidential material referenced in the following set of elements. Requirements for developer actions are identified by appending the letter “D” to the element number.

b) Content and presentation of evidence elements: the evidence required, what the evidence shall demonstrate, and what information the evidence shall convey. Requirements for content and presentation of evidence are identified by appending the letter “C” to the element number.

c) Evaluator action elements: the analysis implied by the evidence provided. This set of actions necessarily includes confirmation that the evidence meets the requirements prescribed in the previous set of elements, and can include actions or analysis which shall be performed in addition to that already performed by the developer. Requirements for evaluator actions are identified by appending the letter “E” to the element number.

50 The developer actions, and content and presentation of evidence define the assurance requirements that are used to represent a developer’s responsibilities in demonstrating assurance in the TOE security functions. By addressing these requirements, the developer can increase confidence that the TOE satisfies the functional and assurance requirements of a PP or a ST.

51 The evaluator actions defines the assurance requirements that represent a TOE evaluator’s responsibilities in verifying the security claims made in the TOE’s ST. By addressing these requirements, the evaluator can increase confidence that the TOE satisfies the functional and assurance requirements of a PP or a ST.

52 These requirements combined with those in the content and presentation of evidence identify the evaluator effort that shall be expended in verifying the security claims made in the ST of the TOE.

53 Confidence that the security claims have been sufficiently verified depends on two factors: the evaluator's competence and objectivity. Although a very important factor of any evaluation, evaluator competence and objectivity are outside the scope of the CC.

2.1.5 Assurance elements

54 Each element represents a requirement to be met. These statements of requirements are intended to be clear, concise, and unambiguous statements. Therefore, there are no compound sentences: each separable requirement is stated as an individual element.

55 The elements have been written using the normal dictionary meaning for the terms used, rather than using a number of predefined terms as shorthand which results in implicit requirements. Therefore, elements are written as explicit requirements, with *no reserved terms*.

2.1.6 EAL structure

56 Figure 2.4 illustrates the evaluation assurance levels and associated structure defined in Part 3 of the CC. Note that while the figure shows the contents of the assurance components, it is intended that this information would be included by reference to the actual components defined in the CC.

2.1.6.1 EAL name

57 Each EAL is assigned a unique name. The name provides descriptive information about the overall thrust of the EAL.

58 A unique short form of the EAL name is also provided. This is the primary means used to reference the EAL.

2.1.6.2 Objectives

59 The objectives section of the EAL provides the overall purpose and goals of the EAL.

2.1.6.3 Application notes

60 The application notes section of the EAL, if present, contains information of particular interest to users of the EAL (e.g., PP and ST authors, designers of TOEs targeting this EAL, evaluators). The presentation is informal and covers, for example, warnings about limitations of use and areas where specific attention may be required.

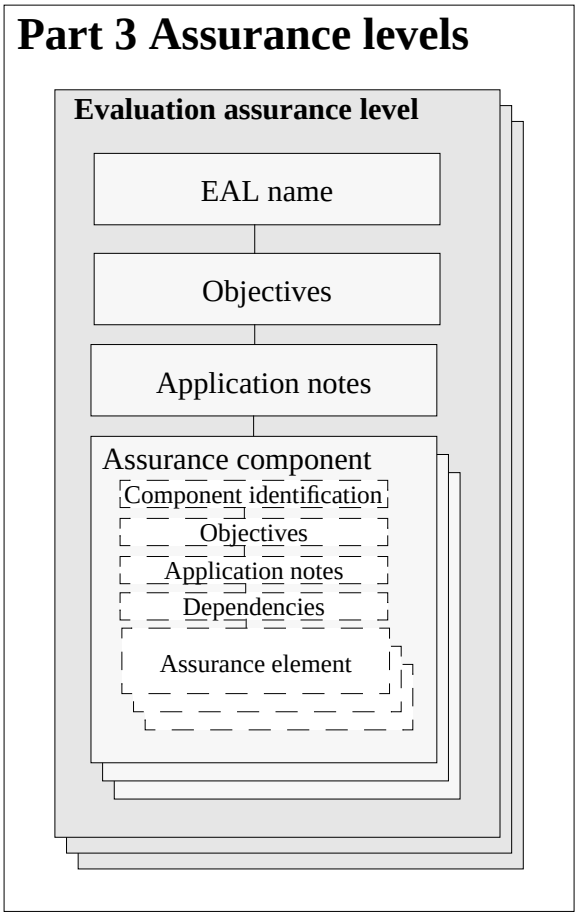

Figure 2.3 - EAL structure

2.1.6.4 Assurance components

61 For each EAL the appropriate assurance components have been chosen.

62 A higher level of assurance can be achieved by:

a) including additional assurance components from other assurance families; or

b) replacing the same assurance component with a higher level assurance component from an assurance family.

2.1.7 Relationship between assurances and assurance levels

63 Figure 2.4 illustrates the relationship between the assurance requirements and the assurance levels defined in Part 3. While assurance components further decompose into assurance elements, assurance elements cannot be individually referenced by

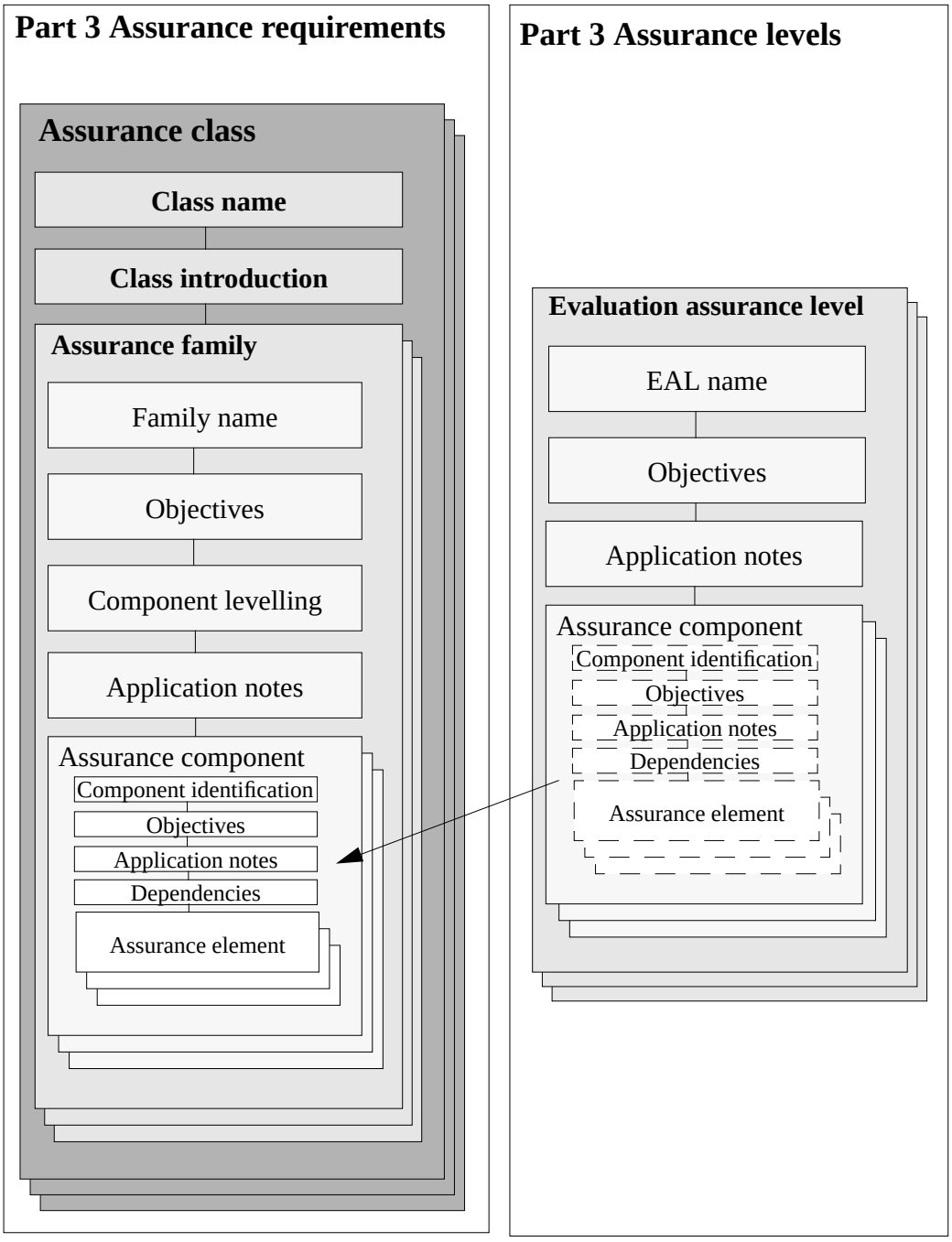

Figure 2.4 - Assurance and assurance level association

assurance levels. Note that the arrow in the figure represents a reference from an EAL to an assurance component within the class where it is defined.

2.2 Component taxonomy

64 Part 3 of the CC contains classes of families and components which are grouped on the basis of related assurance. At the start of each class is a diagram which indicates the families in the class and the components in each family.

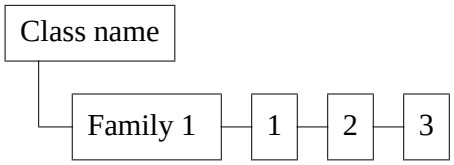

Figure 2.5 - Sample class decomposition diagram

65 In Figure 2.5, above, the class as shown contains a single family. The family contains three components that are linearly hierarchical (i.e., component 2 requires more than component 1, in terms of specific actions, specific evidence, or rigor of the actions or evidence). Unlike the functional families of Part 2, the assurance families are all linearly hierarchical. This is not out of necessity, nor should it direct future development of assurance families, but rather is a result of the derivation of requirements from the source criteria.

2.3 Assurance categorisation

66 The assurance classes, families, and the abbreviation for each family are shown in Table 2.1.

2.4 Assurance class and family overview

67 The following summarises the assurance classes and families of Chapter 5. These classes and family summaries are presented in the same order as they appear in Chapter 5.

2.4.1 Configuration management (ACM)

68 Configuration management (CM) requires that the integrity of the TOE is adequately preserved. Specifically, configuration management provides confidence that the TOE and documentation used for evaluation are the ones prepared for distribution.

2.4.1.1 CM automation (ACM_AUT)

69 Configuration management automation establishes the level of automation used to control the configuration items.

Assurance Class	Assurance Family	Abbreviated Name
Configuration management	CM automation	ACM_AUT
	CM capabilities	ACM_CAP
	CM scope	ACM_SCP
Delivery and operation	Delivery	ADO_DEL
	Installation, generation, and start-up	ADO_IGS
Development	Functional specification	ADV_FSP
	High-level design	ADV_HLD
	Implementation representation	ADV_IMP
	TSF internals	ADV_INT
	Low-level design	ADV_LLD
	Representation correspondence	ADV_RCR
Guidance documents	Administrator guidance	AGD_ADM
	User guidance	AGD_USR
Life cycle support	Development security	ALC_DVS
	Flaw remediation	ALC_FLR
	Life cycle definition	ALC_LCD
	Tools and techniques	ALC_TAT
Tests	Coverage	ATE_COV
	Depth	ATE_DPT
	Functional tests	ATE_FUN
	Independent testing	ATE_IND
Vulnerability assessment	Covert channel analysis	AVA_CCA
	Misuse	AVA_MSU
	Strength of TOE security functions	AVA_SOF
	Vulnerability analysis	AVA_VLA

Table 2.1 -Assurance family breakdown and mapping

2.4.1.2 CM capabilities (ACM_CAP)

70 Configuration management capabilities defines the characteristics of the configuration management system.

2.4.1.3 CM scope (ACM_SCP)

71 Configuration management scope indicates the TOE items that need to be controlled by the configuration management system.

2.4.2 Delivery and operation (ADO)

72 Assurance class ADO defines requirements for the measures, procedures, and standards concerned with secure delivery, installation, and operational use of the

TOE, ensuring that the security protection offered by the TOE is not compromised during transfer, installation, start-up, and operation.

2.4.2.1 Delivery (ADO_DEL)

73 Delivery covers the procedures used to maintain security during transfer of the TOE to the user, both on initial delivery and as part of subsequent modification. It includes special procedures or operations required to demonstrate the authenticity of the delivered TOE. Such procedures and measures are the basis for ensuring that the security protection offered by the TOE is not compromised during transfer. While compliance with the delivery requirements cannot be determined when a TOE is evaluated, it is possible to evaluate the procedures that a developer has developed to distribute the TOE to users.

74 This component is intended to counter the possibility that the TOE could be intentionally subverted during shipment from the development environment to the user's site.

2.4.2.2 Installation, generation, and start-up (ADO_IGS)

75 Installation, generation, and start-up requires that the copy of the TOE is configured and activated by the administrator to exhibit the same protection properties as the master copy of the TOE. The installation, generation, and start-up procedures provide confidence that the administrator will be aware of the TOE configuration parameters and how they can affect the TSF.

2.4.3 Development (ADV)

76 Assurance class ADV defines requirements for the stepwise refinement of the TSF from the TOE summary specification in the ST down to the actual implementation. Each of the resulting TSF representations provide information to help the evaluator determine whether the functional requirements of the TOE have been met.

2.4.3.1 Functional specification (ADV_FSP)

77 The functional specification describes the security functions provided by the TOE. The functional specification must be a complete and accurate interpretation of the requirements, and must also be shown to enforce the TOE security policy. The functional specification also details the external interface to the TOE. Users of the TOE are expected to interact with the TSF through this interface.

2.4.3.2 High-level design (ADV_HLD)

78 The high-level design is a top level design specification that refines the TSF functional specification into the major constituent parts of the TSF. The high level design identifies the basic structure of the TSF and the major hardware, firmware, and software elements.

2.4.3.3 Implementation representation (ADV_IMP)

79 The implementation is the final representation of the TSF which can be used to build the TSF without further design refinement.

2.4.3.4 TSF internals (ADV_INT)

80 The TSF internals are a set of requirements that constrain the internal structuring of the TSF.

2.4.3.5 Low-level design (ADV_LLD)

81 The low-level design is a detailed design specification that refines the high-level design into a level of detail that can be used as a basis for programming and/or hardware construction.

2.4.3.6 Representation correspondence (ADV_RCR)

82 The specification correspondence is a demonstration of a mapping between the lowest-level specification available (e.g., implementation) and the ST requirements.

2.4.4 Guidance documents (AGD)

83 Assurance class AGD defines requirements directed at the understandability, coverage and completeness of the operational documentation provided by the developer. This documentation which provides two categories of information, for end users and for administrators, is an important factor in the secure operation of the TOE.

2.4.4.1 Administrator guidance (AGD_ADM)

84 Requirements for administrative guidance help ensure that the environmental constraints are understood by administrators and operators of the TOE. Administrative guidance is the primary means available to the developer for providing the TOE administrators with detailed, accurate information of how to: (1) configure and install the TOE, (2) operate the TOE in a secure manner, (3) make effective use of the TSF privileges and protection functions to control access to administrative functions and TOE data, and (4) avoid pitfalls and improper use of the administrative functions that would compromise the TSF and user security.

2.4.4.2 User guidance (AGD_USR)

85 Requirements for user guidance help ensure that users are able to operate the TOE in a secure manner (e.g., the usage constraints assumed by the PP must be clearly explained and illustrated). User guidance is the primary vehicle available to the developer for providing the TOE users with the necessary background and specific information on how to correctly use the TOE's protection functions. User guidance must do two things. First, it needs to explain how the user-visible security functions

work, so that users are able to consistently and effectively protect their information. Second, it needs to explain the user's role in maintaining the TOE's security.

2.4.5 Life cycle support (ALC)

86 Assurance class ALC defines requirements for assurance provided in the security of the TOE by the adoption of a well defined life-cycle model for all the steps of the TOE development, including flaw remediation procedures and policies, correct use of tools and techniques and the security measures used to protect the development environment.

2.4.5.1 Development security (ALC_DVS)

87 Development security covers the physical, procedural, personnel, and other security measures used in the development environment. It includes the physical security of the development location(s), and controls on the selection and hiring of development staff.

2.4.5.2 Flaw remediation (ALC_FLR)

88 A part of life cycle support is flaw remediation. Flaw remediation ensures that flaws discovered by the TOE consumers will be tracked and corrected while the TOE is supported by the developer. While compliance with the flaw remediation requirements cannot be determined when a TOE is evaluated, it is possible to evaluate the procedures and policies that a developer has in place to track and repair flaws, and to distribute the repairs to consumers.

2.4.5.3 Life cycle definition (ALC_LCD)

89 Life cycle definition establishes that the engineering practices used by a developer to produce the TOE include the considerations and activities identified in the development process and operational support requirements. Confidence in the correspondence between the requirements and the TOE is greater when security analysis and the production of evidence are done on a regular basis as an integral part of the development process and operational support activities. It is not the intent of this component to dictate any specific development process.

2.4.5.4 Tools and techniques (ALC_TAT)

90 Tools and techniques addresses the need to define the development tools being used to analyse and implement the TOE. It includes requirements concerning the implementation tools and implementation dependent options of those tools.

2.4.6 Tests (ATE)

91 Assurance class ATE states requirements for testing that demonstrate that the TSF satisfies at least the security functional requirements.

2.4.6.1 Coverage (ATE_COV)

92 Coverage deals with the completeness of the functional tests performed on the TOE. It addresses the extent to which the TOE security functions are tested.

2.4.6.2 Depth (ATE_DPT)

93 Depth deals with the level of detail to which the TOE is tested. Testing of security functions is based upon increasing depth of information derived from analysis of the representations.

2.4.6.3 Functional tests (ATE_FUN)

94 Functional testing establishes that the TSF exhibits the properties necessary to satisfy the requirements of its PP and ST. Functional testing provides assurance that the TSF satisfies at least the requirements of the chosen functional components. However, functional tests do not establish that the TSF does no more than expected.

2.4.6.4 Independent testing (ATE_IND)

95 Independent testing specifies the degree to which the functional testing of the TOE must be performed by a party other than the developer (e.g., a third party). This family adds value by the introduction of tests that are not part of the developers tests.

2.4.7 Vulnerability assessment (AVA)

96 Assurance class AVA defines requirements directed at the identification of exploitable vulnerabilities. Specifically, it addresses those vulnerabilities introduced in the construction, operation, misuse, or incorrect configuration of the TOE.

2.4.7.1 Covert channel analysis (AVA_CCA)

97 Covert channel analysis is directed towards the discovery and analysis of unintended communications channels that can be exploited to violate the intended TSP.

2.4.7.2 Misuse (AVA_MSU)

98 This aspect of vulnerability assessment investigates whether the TOE can be configured or used in a manner that is insecure, but that an administrator or end-user of the TOE would reasonably believe to be secure.

2.4.7.3 Strength of TOE security functions (AVA_SOF)

99 Even if a TOE security function cannot be bypassed, deactivated, or corrupted, it may still be possible to defeat it. For these functions, it is possible to make a claim for the strength of each one. For example, a password mechanism cannot prevent the guessing of unknown passwords, but its strength can be increased by making the

password space larger or decreasing the interval between password changes, effectively making passwords less likely to be guessed.

2.4.7.4 Vulnerability analysis (AVA_VLA)

100 This analysis of the TSF consists of the identification of flaws potentially introduced in the different refinement steps of the development. It results in the definition of penetration tests through the collection of the necessary information concerning: (1) the completeness of the TSF (does the TSF counter all the postulated threats?) and (2) the dependencies between all security functions. These known vulnerabilities are assessed through penetration testing to determine whether they could, in practice, be exploitable to compromise the security of the TOE.

Chapter 3

Protection Profile and Security Target evaluation criteria

3.1 Overview

101 This chapter presents the evaluation criteria for PPs and STs. The PP evaluation criteria are presented in the "Class APE" and the ST evaluation criteria are presented in the "Class ASE".

102 These criteria are the first requirements presented in this part because the PP and ST evaluation will normally be performed before the TOE evaluation. They play a special role insofar that information about the TOE is assessed and the functional and assurance requirements are evaluated in order to find out whether the PP or ST is a meaningful basis for a TOE evaluation.

103 Although these evaluation criteria differ somewhat from the requirements in Chapter 5, they are presented in a similar manner because the developer and evaluator activities are comparable for the PP evaluation, the ST evaluation and the TOE evaluation proper.

104 The classes in this chapter differ from those in Chapter 5 in that all requirements in the respective class need to be applied for a PP or ST evaluation, whereas the requirements presented in Chapter 5 allow selection.

3.2 Protection Profile criteria overview

3.2.1 Protection Profile evaluation

105 The goal of a PP evaluation is to demonstrate that the PP is complete, consistent, technically sound, and hence suitable for use as a statement of requirements for an evaluatable TOE. Such a PP may be eligible for inclusion within a PP register.

3.2.2 Relation to the Security Target evaluation criteria

106 As described in Part 1, Annexes B and C, there are many similarities in structure and content between the generic PP and the TOE-specific ST. Consequently, the criteria for evaluating PPs contain requirements that are similar to many of those for STs, and the criteria for both are presented in a similar manner.

3.2.3 Evaluator tasks

107 Evaluators performing a PP evaluation shall apply all requirements of the APE class as described in Table 3.1.

-

Class	Family	Abbreviated Name
Protection Profile evaluation	Protection Profile, Security Environment	APE_ENV
	Protection Profile, Security Objectives	APE_OBJ
	Protection Profile, TOE Security Requirements	APE_REQ

Table 3.1 - Protection Profile families

3.3 Security Target criteria overview

3.3.1 Security Target evaluation

108 The goal of a ST evaluation is to demonstrate that the ST is complete, consistent, technically sound, and hence suitable for use as the basis for the corresponding TOE evaluation.

3.3.2 Relation to the other evaluation criteria in this Part

109 The TOE evaluation according to the definition in Part 1, 2.3.5 b) contains two identified stages, the ST evaluation and the TOE evaluation. The requirements for the ST evaluation are contained in this chapter, and the requirements for the TOE evaluation are contained in Chapters 4 and 5.

110 The ST evaluation includes a PP claims evaluation. If the ST does not claim PP conformance, the PP claims part of the ST shall contain a statement that the TOE does not claim conformance to any PP.

3.3.3 Evaluator tasks

111 Evaluators performing a ST evaluation shall apply all requirements of the ASE class as described in Table 3.1.

Class	Family	Abbreviated Name
Security Target evaluation	Security Target, Security Environment	ASE_ENV
	Security Target, Security Objectives	ASE_OBJ
	Security Target, PP Claims	ASE_PPC
	Security Target, TOE Security Requirements	ASE_REQ
	Security Target, TOE Summary Specification	ASE_TSS

Table 3.2 -Security Target families

Class APE

Protection Profile evaluation

112 The Protection Profile Evaluation confirms that the PP represents a meaningful TOE with a consistent security policy and is suitable for inclusion in a registry.

113 Figure 3.1 shows the families within this class.

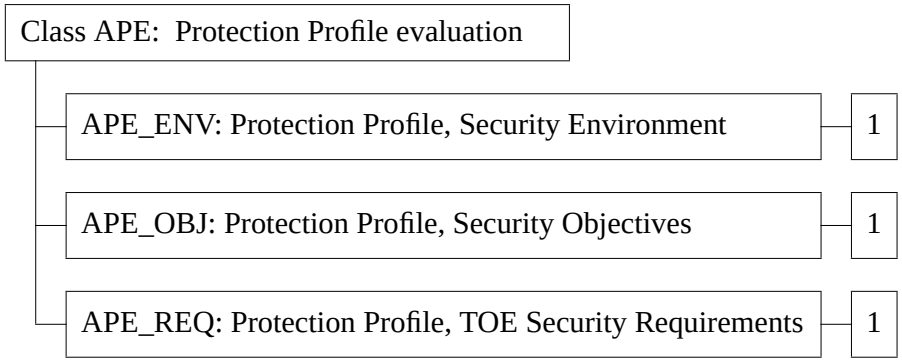

Figure 3.1 - Protection Profile evaluation class decomposition

APE_ENV Protection Profile, Security Environment

Objectives

114 In order to determine whether the IT security requirements in the PP are sufficient, it is important that the security problem to be solved is clearly understood by all parties to the evaluation.

Application notes

115 The contents of a PP are portrayed in Part 1, Annex B, Figure B.1, which shows a general structure that should be used for presentation.

116 The evaluation of a PP shall ensure that the structure for the PP is supportive of the evaluation process.

APE_ENV.1 Protection Profile, Security Environment, Evaluation Requirements

Dependencies:

No dependencies.

Developer action elements:

APE_ENV.1.1D **The PP developer shall provide a statement of TOE security environment as part of the PP.**

Content and presentation of evidence elements:

APE_ENV.1.1C **The statement of TOE security environment shall identify and explain any known or presumed threats to the IT assets against which protection will be required, either by the TOE or by the security environment.**

APE_ENV.1.2C **The statement of TOE security environment shall identify and explain any organisational security policies that the TOE must comply with.**

APE_ENV.1.3C **The statement of TOE security environment shall identify and explain any assumptions about the secure usage of the TOE in its anticipated or actual environment of use. These include, but are not limited to, assumptions regarding the physical, personnel, and connectivity aspects of that environment.**

Evaluator action elements:

APE_ENV.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

APE_ENV.1.2E **The evaluator shall confirm that the statement of TOE security environment is complete, coherent, and internally consistent.**

APE_OBJ Protection Profile, Security Objectives

Objectives

117 The security objectives is a concise statement of the intended response to the security problem. Evaluation of the security objectives is required to demonstrate that the stated objectives adequately address the security problem. The security objectives are categorised as IT security objectives and non-IT security objectives. The security objectives must be shown to be traced back to the identified threats to be countered and/or policies which are to be met by the TOE.

APE_OBJ.1 Protection Profile, Security Objectives, Evaluation Requirements

Dependencies:

APE_ENV.1 Protection Profile, Security Environment, Evaluation Requirements

Developer action elements:

APE_OBJ.1.1D **The PP developer shall provide a statement of security objectives as part of the PP.**

APE_OBJ.1.2D **The PP developer shall provide the rationale for the security objectives.**

Content and presentation of evidence elements:

APE_OBJ.1.1C **The statement of security objectives shall distinguish between the IT security objectives and any relevant non-IT security objectives.**

APE_OBJ.1.2C **The IT security objectives shall be clearly stated and traced back to the identified threats to be countered and/or organisational security policies to be met by the IT.**

APE_OBJ.1.3C **Any non-IT security objectives shall be clearly stated and traced back to the identified threats to be countered and/or organisational security policies to be met by the non-IT environmental measures.**

APE_OBJ.1.4C **An argument shall be provided that demonstrates that the stated security objectives counter all of the identified threats to security.**

APE_OBJ.1.5C **A description shall be provided that demonstrates that all of the identified threats to be countered are addressed by and all security policies are met by one or more security objective.**

Evaluator action elements:

APE_OBJ.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

APE_OBJ.1.2E **The evaluator shall confirm that the statement of security objectives is complete, coherent, and internally consistent.**

APE_REQ Protection Profile, TOE Security Requirements

Objectives

118 The IT security requirements chosen for a TOE and presented or cited in a PP need to be evaluated in order to demonstrate that they are internally self consistent and lead to the development of a TOE which will meet its security objectives.

119 Not all of the IT security objectives expressed in a PP may be met by a compliant TOE. So some TOEs may depend on certain IT security requirements to be met by the IT environment. When this is the case, the environmental IT security requirements must be clearly stated and evaluated in context with the TOE requirements.

120 This family presents evaluation requirements which permit the evaluator to determine that a TOE compliant with its requirements meets its objectives.

APE_REQ.1 Protection Profile, TOE Security Requirements, Evaluation Requirements

Dependencies:

APE_OBJ.1 Protection Profile, Security Objectives, Evaluation Requirements

Developer action elements:

APE_REQ.1.1D The PP developer shall provide a statement of TOE IT functional requirements and a statement of TOE IT assurance requirements as part of the PP.

APE_REQ.1.2D The PP developer shall provide the rationale for the IT security requirements.

Content and presentation of evidence elements:

APE_REQ.1.1C TOE IT functional requirements shall be expressed using CC Part 2 functional requirements components.

APE_REQ.1.2C TOE IT assurance requirements shall be expressed using CC Part 3 assurance requirements components.

APE_REQ.1.3C TOE IT assurance requirements shall include a CC evaluation assurance level as defined in CC Part 3.

APE_REQ.1.4C The PP shall, if necessary, identify and define any security requirements for the IT environment.

APE_REQ.1.5C Security requirements for the IT environment shall be stated by reference to CC Part 2 or 3 security requirements where it is feasible to do so.

APE_REQ.1.6C **All operations on CC security requirements included in the PP shall be identified and explained.**

APE_REQ.1.7C **Any uncompleted operations shall be clearly identified and described.**

APE_REQ.1.8C **Dependencies of CC security requirements included in the PP shall be accounted for and shown to be satisfied.**

APE_REQ.1.9C **The PP shall demonstrate that the articulated security requirements are suitable to meet all of the TOE IT security objectives.**

APE_REQ.1.10C **The PP shall demonstrate that the set of IT security requirements together forms a mutually supportive and internally consistent whole.**

Evaluator action elements:

APE_REQ.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

APE_REQ.1.2E **The evaluator shall confirm that the set of IT security requirements is suitable to meet all of the IT security objectives.**

Class ASE

Security Target evaluation

121 The goal of a ST evaluation is to demonstrate that the ST is complete, consistent, technically sound, and hence suitable for use as the basis for the corresponding TOE evaluation.

122 Figure 3.2 shows the families within this class.

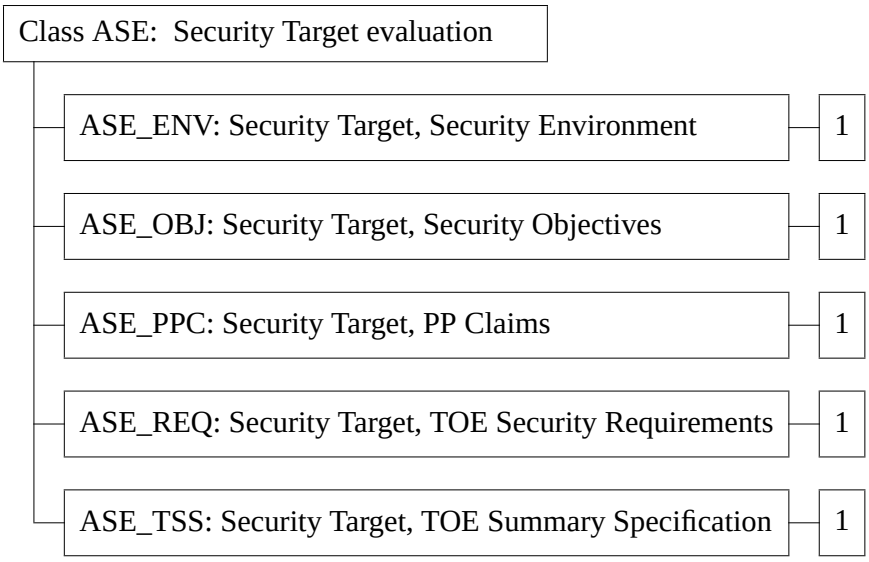

Figure 3.2 - Security Target evaluation class decomposition

ASE_ENV Security Target, Security Environment

Objectives

123 In order to determine whether the IT security requirements in the ST are sufficient, it is important that the security problem to be solved is clearly understood by all parties to the evaluation.

Application notes

124 The contents of a ST are portrayed in Part 1, Annex C, Figure C.1, which shows a general structure that should be used for presentation.

125 The evaluation of a ST shall ensure that the structure for the ST is supportive of the evaluation process.

ASE_ENV.1 Security Target, Security Environment, Evaluation Requirements

Dependencies:

No dependencies.

Developer action elements:

ASE_ENV.1.1D **The developer shall provide a statement of TOE security environment as part of the ST.**

Content and presentation of evidence elements:

ASE_ENV.1.1C **The statement of TOE security environment shall identify and explain any known or presumed threats to the IT assets against which protection will be required, either by the TOE or by the security environment.**

ASE_ENV.1.2C **The statement of TOE security environment shall identify and explain any organisational security policies that the TOE must comply with.**

ASE_ENV.1.3C **The statement of TOE security environment shall identify and explain any assumptions about the secure usage of the TOE in its anticipated or actual environment of use. These include, but are not limited to, assumptions regarding the physical, personnel, and connectivity aspects of that environment.**

Evaluator action elements:

ASE_ENV.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ASE_ENV.1.2E **The evaluator shall confirm that the statement of TOE security environment is complete, coherent, and internally consistent.**

ASE_OBJ Security Target, Security Objectives

Objectives

126 The security objectives are a concise statement of the intended response to the security problem. Evaluation of the security objectives is required to demonstrate that the stated objectives adequately address the security problem. The security objectives are categorised as IT security objectives and non-IT security objectives. The security objectives must be shown to be traced back to the identified threats to be countered and/or policies which are to be met by the TOE.

ASE_OBJ.1 Security Target, Security Objectives, Evaluation Requirements

Dependencies:

ASE_ENV.1 Security Target, Security Environment, Evaluation Requirements

Developer action elements:

ASE_OBJ.1.1D **The developer shall provide a statement of security objectives as part of the ST.**

ASE_OBJ.1.2D **The developer shall provide the rationale for the security objectives.**

Content and presentation of evidence elements:

ASE_OBJ.1.1C **The statement of security objectives shall distinguish between the IT security objectives and any relevant non-IT security objectives.**

ASE_OBJ.1.2C **The IT security objectives shall be clearly stated and traced back to the identified threats to be countered and/or organisational security policies to be met by the IT.**

ASE_OBJ.1.3C **Any non-IT security objectives shall be clearly stated and traced back to the identified threats to be countered and/or organisational security policies to be met by the non-IT environmental measures**

ASE_OBJ.1.4C **An argument shall be provided that demonstrates that the stated security objectives counter all of the identified threats to security.**

ASE_OBJ.1.5C **A description shall be provided that demonstrates that all of the identified threats to be countered are addressed by and all security policies are met by one or more security objective.**

Evaluator action elements:

ASE_OBJ.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ASE_OBJ.1.2E **The evaluator shall confirm that the statement of security objectives is complete, coherent, and internally consistent.**

ASE_PPC Security Target, PP Claims

Objectives

127 The goal of the evaluation of the Security Target PP claims is to determine whether the ST is a correct instantiation of the PP.

Application notes

128 The family includes the case that no PP claim is made as part of the ST. In this case there is no need for any evaluation activities to be performed by the evaluator.

129 Although additional evaluation activity is necessary when a PP claim is made, the ST evaluation effort is generally smaller than in cases where no PP is used because it is possible to reuse the PP evaluation results for the ST evaluation.

ASE_PPC.1 Security Target PP Claims Evaluation Requirements

Dependencies:

ASE_OBJ.1 Security Target, Security Objectives, Evaluation Requirements

ASE_REQ.1 Security Target, TOE Security Requirements, Evaluation Requirements

Developer action elements:

ASE_PPC.1.1D **The developer shall provide any PP claims as part of the ST.**

ASE_PPC.1.2D **The developer shall provide the PP rationale for each provided PP claim.**

Content and presentation of evidence elements:

ASE_PPC.1.1C **Each PP claim shall identify the PP for which compliance is being claimed, including qualifications needed for that claim.**

ASE_PPC.1.2C **Each PP claim shall identify the IT security objectives and requirements statements which satisfy the permitted operations of the PP or otherwise further refine the PP's IT security objectives and requirements.**

ASE_PPC.1.3C **Each PP claim shall identify IT security objectives and requirements statements which are additional to PP objectives and requirements.**

ASE_PPC.1.4C **Each PP claim shall demonstrate that the totality of the IT security requirements statements include, support, and do not conflict with any PP requirements.**

ASE_PPC.1.5C **Each PP claim shall demonstrate that any refinements of PP objectives result in valid interpretations of the more abstract statements of objectives within that PP.**

ASE_PPC.1.6C **Each PP claim shall demonstrate that any refinements of and operations on PP requirements result in valid interpretations of the more abstract requirements within that PP.**

ASE_PPC.1.7C **Each PP claim shall demonstrate that all PP objectives are met and requirements are satisfied.**

Evaluator action elements:

ASE_PPC.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ASE_PPC.1.2E **The evaluator shall confirm that the PP claims are complete, coherent, and internally consistent.**

ASE_REQ Security Target, TOE Security Requirements

Objectives

130 The IT security requirements chosen for a TOE and presented or cited in a ST need to be evaluated in order to confirm that they are internally self consistent and lead to the development of a TOE which will meet its security objectives.

131 This family presents evaluation requirements which permit the evaluator to determine that a TOE compliant with its requirements meets its objectives.

Application notes

132 In Part 1, Annex C, Section C.2.5 it is outlined that if none of the Part 2 functional requirements components are readily applicable to all or parts of the TOE security requirements, the ST may state those functional requirements explicitly without reference to the CC. The ST may also extend the EAL by stating additional assurance requirements not taken from Part 3.

133 Consequently, in this family some elements specify that requirements shall be expressed using Parts 2 and 3 where it is feasible to do so.

ASE_REQ.1 Security Target, TOE Security Requirements, Evaluation Requirements

Dependencies:

ASE_OBJ.1 Security Target, Security Objectives, Evaluation Requirements

Developer action elements:

ASE_REQ.1.1D The developer shall provide a statement of TOE IT functional requirements and a statement of TOE IT assurance requirements as part of the ST.

ASE_REQ.1.2D The developer shall provide the rationale for the IT security requirements.

Content and presentation of evidence elements:

ASE_REQ.1.1C TOE IT functional requirements shall be expressed using CC Part 2 functional requirements components where it is feasible to do so.

ASE_REQ.1.2C TOE IT assurance requirements shall be expressed using CC Part 3 assurance requirements components where it is feasible to do so.

ASE_REQ.1.3C TOE IT assurance requirements shall include a CC evaluation assurance level as defined in CC Part 3.

ASE_REQ.1.4C The ST shall, if necessary, identify and define any security requirement for the IT environment.

ASE_REQ.1.5C **Security requirements for the IT environment shall be stated using CC Parts 2 or 3 security requirements where it is feasible to do so.**

ASE_REQ.1.6C **Security requirements which do not reference CC components shall use the CC requirements components as a model for presentation.**

ASE_REQ.1.7C **The ST shall demonstrate that any explicitly stated security requirements identify and satisfy the objectives which they are intended to meet.**

ASE_REQ.1.8C **The ST shall demonstrate that the assurance requirements are applicable and appropriate to support any explicitly stated functional requirements.**

ASE_REQ.1.9C **Operations on CC security requirements included in the ST shall be identified, explained, and completed.**

ASE_REQ.1.10C **Dependencies of CC security requirements included in the ST shall be accounted for and shown to be satisfied.**

ASE_REQ.1.11C **All security requirements shall be measurable and state objective evaluation requirements such that compliance or noncompliance of a TOE can be determined and systematically demonstrated.**

ASE_REQ.1.12C **All IT security requirements shall be expressed unambiguously in order to eliminate the need for interpretation.**

ASE_REQ.1.13C **All IT security requirements shall be traced to the IT security objectives such that it can be seen which IT security requirement satisfies which IT security objective and that every IT security requirement contributes to the satisfaction of at least one IT security objective.**

ASE_REQ.1.14C **The ST shall demonstrate that the articulated security requirements are suitable to meet the TOE IT security objectives.**

ASE_REQ.1.15C **The ST shall demonstrate that the set of IT security requirements together forms a mutually supportive and internally consistent whole.**

Evaluator action elements:

ASE_REQ.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ASE_REQ.1.2E **The evaluator shall confirm that the set of IT security requirements is suitable to meet all of the IT security objectives.**

ASE_TSS Security Target, TOE Summary Specification

Objectives

134 The TOE summary specification provides a high-level definition of the security functions claimed to meet the functional requirements and of the assurance measures taken to meet the assurance requirements.

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

Dependencies:

ASE_OBJ.1 Security Target, Security Objectives, Evaluation Requirements

ASE_REQ.1 Security Target, TOE Security Requirements, Evaluation Requirements

Developer action elements:

ASE_TSS.1.1D **The developer shall provide a TOE summary specification as part of the ST.**

ASE_TSS.1.2D **The developer shall provide the rationale for the TOE summary specification.**

Content and presentation of evidence elements:

ASE_TSS.1.1C **The TOE summary specification shall define the TOE IT security functions and the TOE assurance measures.**

ASE_TSS.1.2C **The TOE summary specification shall trace the security functions to the security requirements such that it can be seen which TOE IT security functions satisfy which requirements and that every security function contributes to the satisfaction of at least one security requirement.**

ASE_TSS.1.3C **The TOE IT security functions shall be defined in an informal style to a level of detail necessary for understanding of the intent and behaviour of the function.**

ASE_TSS.1.4C **Any references to security mechanisms and techniques included in the ST shall be traced to the relevant security functions so that it can be seen which mechanisms or techniques are used in the implementation of each function.**

ASE_TSS.1.5C **The TOE summary specification shall demonstrate that the TOE IT security functions meet all functional requirements of the TOE.**

ASE_TSS.1.6C **The TOE summary specification shall demonstrate that the combination of the specified TOE IT security functions work together in a mutually supportive manner in order to satisfy the TOE security objectives.**

ASE_TSS.1.7C **The TOE summary specification shall trace the assurance measures to the assurance requirements such that it can be seen which measures satisfy which requirements.**

ASE_TSS.1.8C **The TOE summary specification shall demonstrate that the assurance measures meet all assurance requirements of the TOE.**

Evaluator action elements:

ASE_TSS.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ASE_TSS.1.2E **The evaluator shall confirm that the TOE summary specification is complete, coherent, and internally consistent.**

Chapter 4

Assurance levels

135 The Evaluation Assurance Levels (EALs) provide a uniformly increasing scale which balances the level of assurance obtained with the cost and feasibility of acquiring that degree of assurance.

136 While the CC has adopted the evaluation-based criteria philosophy of its predecessors, the EALs were developed within that philosophy but with a different scope. The CC approach divides the concepts of assurance in a TOE at the end of the evaluation and maintenance of that assurance during the operational use of the TOE. The result being a departure from the evaluation levels of the various predecessors of the CC inasmuch as some of the assurance families are not included in any EAL.

137 In defining the EALs, an analysis was performed which concluded that every assurance family, except “Delivery” and “Flaw remediation”, contributes directly to the assurance that a TOE meets its security claims at the end of the evaluation. As the assurance paradigm is based on assurance gained during evaluation, the EALs are based on those assurance families. This is supported by the fact that evaluators gain “real” assurance by the first hand application of assurance mechanisms (e.g., analysis and testing of an existing design), while they can gain only “theoretical” assurance for mechanisms applied after the evaluation (e.g., a plan for delivery of the TOE). In other words, while such assurance mechanisms can be evaluated to determine whether they can provide their claimed assurance, it is not possible to produce practical evidence of their future application.

138 It is important to note that the “Delivery” and “Flaw remediation” families, as well as some aspects of the other families (e.g., “CM capabilities”), can be evaluated and provide meaningful and desired assurances. The assurance that they provide contributes to maintaining that initial assurance determined by the evaluation of the TOE. Note that while these families are not specifically included in any EAL, it is expected and recommended that they be considered for augmentation of an EAL in PPs and STs.

4.1 Evaluation assurance level (EAL) overview

Assurance Class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Configuration management	ACM_AUT				**1**	1	**2**	2
	ACM_CAP	**1**	1	**2**	**3**	3	**4**	4
	ACM_SCP			**1**	**2**	**3**	3	3
Delivery and operation	ADO_DEL							
	ADO_IGS		**1**	1	1	1	1	1
Development	ADV_FSP	**1**	1	1	**2**	**4**	**5**	**6**
	ADV_HLD		**1**	**2**	2	**3**	**4**	**5**
	ADV_IMP				**1**	**2**	**3**	3
	ADV_INT					**1**	**2**	**3**
	ADV_LLD				**1**	1	**2**	2
	ADV_RCR	**1**	1	1	1	**2**	2	**3**
Guidance documents	AGD_ADM	**1**	1	1	1	1	1	1
	AGD_USR	**1**	1	1	1	1	1	1
Life cycle support	ALC_DVS			**1**	1	1	**2**	2
	ALC_FLR							
	ALC_LCD				**1**	**2**	2	**3**
	ALC_TAT				**1**	**2**	**3**	3
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT		**1**	**2**	2	**3**	3	**4**
	ATE_FUN		**1**	1	1	1	1	1
	ATE_IND	**1**	1	**2**	2	2	2	**3**
Vulnerability assessment	AVA_CCA					**1**	**2**	2
	AVA_MSU			**1**	**2**	2	2	2
	AVA_SOF		**1**	1	1	1	1	1
	AVA_VLA		**1**	1	**2**	**3**	**4**	4

Table 4.1 -Evaluation Assurance Level Summary

139 Table 4.1 represents a summary of the EALs. The columns represent a hierarchically ordered set of EALs, while the rows represent assurance families. Each point in the resulting matrix identifies a specific assurance component where applicable.

140 As outlined in the next section, seven hierarchically ordered evaluation assurance levels that can be selected are defined in this CC for the rating of the TOE's assurance. They are hierarchically ordered inasmuch as each EAL represents more assurance than all lower EALs. The increase in assurance from EAL to EAL is

accomplished by *substituting* a hierarchically higher assurance component from the same assurance family (i.e., increasing rigour, scope, and/or depth) and from the *addition* of assurance components from other assurance families (i.e., adding new requirements).

141 These EALs consist of an appropriate combination of assurance components as described in Chapter 2 of this Part. More precisely, each EAL includes no more than one component of each assurance family and all assurance dependencies of every component are addressed.

142 While the EALs are defined in the CC, it is possible to represent other combinations of assurance. Specifically, the notion of "augmentation" allows the addition of assurance components (from assurance families not already included in the EAL) or the substitution of assurance components (with another hierarchically higher assurance component in the same assurance family) to an EAL. Of the assurance constructs defined in the CC, only EALs may be augmented. Furthermore, an EAL may be altered only be augmentation. The notion of an "EAL minus a constituent assurance component" is not recognised by the CC as a valid claim. Augmentation carries with it the obligation on the part of the claimant to justify the utility and added value of the added assurance component to the EAL.

4.2 Evaluation assurance level details

143 The following sections provide definitions of the EALs, highlighting differences between the specific requirements and the prose characterisations of those requirements using bold type.

4.2.1 Evaluation assurance level 1 (EAL1) - functionally tested

Objectives

144 EAL1 is the lowest assurance level for which evaluation is meaningful and economically justified. EAL1 is intended to detect obvious errors for a minimum outlay but is unlikely to result in the detection of other than very obvious security weaknesses.

145 EAL1 is applicable in circumstances where those responsible for user data may wish or be obliged to seek independent assurances in the IT security but the risks to security are not viewed as serious. Under these circumstances, an EAL1 rating would be of value to support the contention that due care had been exercised with respect to personal or similar information.

Assurance components

146 **EAL1 (see Table 4.2) provides a minimum level of assurance by an analysis of the security functions using a functional and interface specification of the TOE, to understand the security behaviour.**

147 **The analysis is supported by independent testing of each of the security functions.**

148 This EAL, nonetheless, represents a meaningful increase over an un-evaluated IT product or system (TOE).

Assurance class	Assurance components
Configuration management	**ACM_CAP.1 Minimal support**
Development	**ADV_FSP.1 TOE and security policy**
	ADV_RCR.1 Informal correspondence demonstration
Guidance documents	**AGD_ADM.1 Administrator guidance**
	AGD_USR.1 User guidance
Tests	**ATE_IND.1 Independent testing - conformance**

Table 4.2 -EAL1

4.2.2 Evaluation assurance level 2 (EAL2) - structurally tested

Objectives

149 EAL2 is the highest assurance level that can be used without imposing other than minimal additional tasks upon the developer. If the developer applies reasonable standards of care to the development, EAL2 may be feasible without developer involvement other than support for security functional testing.

150 EAL2 is therefore applicable in those circumstances where developers or users require a low to moderate level of independently assured security in the absence of ready availability of the complete development record. Such a situation may arise when securing legacy systems or where access to the developer may be limited.

Assurance components

151 **EAL2** (see Table **4.3**) provides assurance by an analysis of the security functions using a functional and interface specification **and the high-level design of the subsystems** of the TOE, to understand the security behaviour**.**

152 The analysis is supported by independent testing of each of the security functions**, evidence of developer "black box" testing, and evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain)**.

153 This EAL represents a meaningful increase in assurance from EAL1 by requiring developer testing, a vulnerability analysis, and independent testing based upon more detailed TOE specifications.

Assurance class	Assurance components
Configuration management	ACM_CAP.1 Minimal support
Delivery and operation	**ADO_IGS.1 Installation, generation, and start-up procedures**
Development	ADV_FSP.1 TOE and security policy
	ADV_HLD.1 Descriptive high-level design
	ADV_RCR.1 Informal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Tests	**ATE_COV.1 Complete coverage - informal**
	ATE_DPT.1 Testing - functional specification
	ATE_FUN.1 Functional testing
	ATE_IND.1 Independent testing - conformance
Vulnerability assessment	**AVA_SOF.1 Strength of TOE security function evaluation**
	AVA_VLA.1 Developer vulnerability analysis

Table 4.3 -EAL2

4.2.3 Evaluation assurance level 3 (EAL3) - methodically tested and checked

Objectives

154 EAL3 permits a conscientious developer to gain maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices.

155 EAL3 is therefore applicable in those circumstances where developers or users require a moderate level of independently assured security and require a thorough investigation of the product and its development without incurring substantial re-engineering costs.

Assurance components

156 **EAL3** (see Table **4.4**) provides assurance by an analysis of the security functions using a functional and interface specification and the high-level design of the subsystems of the TOE, to understand the security behaviour**.**

157 The analysis is supported by independent testing of the security functions, evidence of developer "**gray** box" testing, **selective independent confirmation of the developer test results**, and evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain).

158 **EAL3 also provides added assurance through the addition of development environment controls and TOE configuration management.**

159 This EAL represents a meaningful increase in assurance from EAL2 by requiring more complete testing coverage of the security functions and mechanisms and/or procedures that provide some confidence that the TOE will not be tampered with during development.

Assurance class	Assurance components
Configuration management	**ACM_CAP.2 Authorisation controls**
	ACM_SCP.1 Minimal CM coverage
Delivery and operation	ADO_IGS.1 Installation, generation, and start-up procedures
Development	ADV_FSP.1 TOE and security policy
	ADV_HLD.2 Security enforcing high-level design
	ADV_RCR.1 Informal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	**ALC_DVS.1 Identification of security measures**
Tests	**ATE_COV.2 Complete coverage - rigorous**
	ATE_DPT.2 Testing - high level design
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
Vulnerability assessment	**AVA_MSU.1 Misuse analysis - obvious flaws**
	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.1 Developer vulnerability analysis

Table 4.4 -EAL3

4.2.4 Evaluation assurance level 4 (EAL4) - methodically designed, tested, and reviewed

Objectives

160 EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources. EAL4 is the highest level which it is likely to be economically feasible to retrofit to an existing product line.

161 EAL4 is therefore applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity products and are prepared to incur additional security specific engineering costs.

Assurance components

162 **EAL4** (see Table **4.5**) provides assurance by an analysis of the security functions using a functional and interface specification**,** the high-level design of the subsystems, **the low-level design of the modules of the TOE, and a subset of the implementation**, to understand the security behaviour**.**

163 The analysis is supported by independent testing of the security functions, evidence of developer "gray box" testing, selective independent confirmation of the developer test results, evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain), **and an independent search for obvious vulnerabilities**.

164 **EAL4** also provides assurance through the **use** of development environment controls and **additional** TOE configuration management **including automation**.

165 This EAL represents a meaningful increase in assurance from EAL3 by requiring more design description, a subset of the implementation, and improved mechanisms and/or procedures that provide confidence that the TOE will not be tampered with during development.

Assurance class	Assurance components
Configuration management	**ACM_AUT.1 Partial CM automation**
	ACM_CAP.3 Generation support and acceptance procedures
	ACM_SCP.2 Problem tracking CM coverage
Delivery and operation	ADO_IGS.1 Installation, generation, and start-up procedures
Development	**ADV_FSP.2 Informal security policy model**
	ADV_HLD.2 Security enforcing high-level design
	ADV_IMP.1 Subset of the implementation of the TSF
	ADV_LLD.1 Descriptive low-level design
	ADV_RCR.1 Informal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	**ALC_DVS.1 Identification of security measures**
	ALC_LCD.1 Developer defined life-cycle model
	ALC_TAT.1 Well defined development tools
Tests	ATE_COV.2 Complete coverage - rigorous
	ATE_DPT.2 Testing - high level design
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
Vulnerability assessment	**AVA_MSU.2 Misuse analysis - independent verification**
	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.2 Independent vulnerability analysis

Table 4.5 -EAL4

4.2.5 Evaluation assurance level 5 (EAL5) - semiformally designed and tested

Objectives

166 EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques. Such a product will be designed and developed with the intent of achieving EAL5 assurance. It is likely that the additional costs attributable to the EAL5 requirements relative to rigorous development without the application of specialised techniques will not be excessive.

167 EAL5 is therefore applicable in those circumstances where developers or users require a high level of independently assured security in a planned development and require a rigorous development approach without incurring unreasonable costs attributable to specialist security engineering techniques.

Assurance components

168 **EAL5** (see Table **4.6**) provides assurance by an analysis of the security functions using a functional and interface specification**,** the high-level design of the subsystems, the low-level design of the modules of the TOE, and **all** of the implementation, to understand the security behaviour**. Assurance is additionally gained through a formal model and a semiformal presentation of the functional specification and high-level design and a semiformal demonstration of correspondence between them.**

169 The analysis is supported by independent testing of the security functions, evidence of developer "gray box" testing, selective independent confirmation of the developer test results, evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain), and an independent search for **vulnerabilities ensuring relative resistance to penetration attack**. **The analysis also includes a search for covert channels, when applicable, and is supported by requiring a modular TOE design.**

170 **EAL5** also provides assurance through the use of a development environment controls, and **comprehensive** TOE configuration management including automation.

171 This EAL represents a meaningful increase in assurance from EAL4 by requiring semiformal design descriptions, the entire implementation, a more structured (and hence analysable) architecture, covert channel analysis, and improved mechanisms and/or procedures that provide confidence that the TOE will not be tampered with during development.

Assurance class	Assurance components
Configuration management	ACM_AUT.1 Partial CM automation
	ACM_CAP.3 Generation support and acceptance procedures
	ACM_SCP.3 Development tools CM coverage
Delivery and operation	ADO_IGS.1 Installation, generation, and start-up procedures
Development	**ADV_FSP.4 Formal security policy model**
	ADV_HLD.3 Semiformal high-level design
	ADV_IMP.2 Implementation of the TSF
	ADV_INT.1 Modularity
	ADV_LLD.1 Descriptive low-level design
	ADV_RCR.2 Semiformal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	ALC_DVS.1 Identification of security measures
	ALC_LCD.2 Standardised life-cycle model
	ALC_TAT.2 Compliance with implementation standards
Tests	ATE_COV.2 Complete coverage - rigorous
	ATE_DPT.3 Testing - low level design
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
Vulnerability assessment	**AVA_CCA.1 Covert channel analysis**
	AVA_MSU.2 Misuse analysis - independent verification
	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.3 Relatively resistant

Table 4.6 -EAL5

4.2.6 Evaluation assurance level 6 (EAL6) - semiformally verified design and tested

Objectives

172 EAL6 permits developers to gain high assurance from application of security engineering techniques to a rigorous development environment in order to produce a premium product for protecting high value assets against significant risks.

173 EAL6 is therefore applicable to the development of specialist security products for application in high risk situations where the value of the protected assets justifies the additional costs.

Assurance components

174 **EAL6** (see Table **4.7**) provides assurance by an analysis of the security functions using a functional and interface specification**,** the high-level design of the subsystems, the low-level design of the modules of the TOE, and **a structured presentation** of the implementation, to understand the security behaviour**.** Assurance is additionally gained through a formal model, a semiformal presentation of the functional specification, high-level design**, and low-level design** and a semiformal demonstration of correspondence between them.

175 The analysis is supported by independent testing of the security functions, evidence of developer "gray box" testing, selective independent confirmation of the developer test results, evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain), and an independent search for vulnerabilities ensuring **high** resistance to penetration attack. The analysis also includes a **systematic** search for covert channels, when applicable, and is supported by requiring a modular **and layered** TOE design.

176 **EAL6** also provides assurance through the use of a **structured development process,** development environment controls, and comprehensive TOE configuration management including **complete** automation.

177 This EAL represents a meaningful increase in assurance from EAL5 by requiring more comprehensive analysis, a structured representation of the implementation, more architectural structure (e.g., layering), more comprehensive independent vulnerability analysis, systematic covert channel identification, and improved configuration management and development environment controls.

Assurance class	Assurance components
Configuration management	**ACM_AUT.2 Complete CM automation**
	ACM_CAP.4 Advanced support
	ACM_SCP.3 Development tools CM coverage
Delivery and operation	ADO_IGS.1 Installation, generation, and start-up procedures
Development	**ADV_FSP.5 Property specification by model interpretation**
	ADV_HLD.4 Semiformal high-level explanation
	ADV_IMP.3 Structured implementation of the TSF
	ADV_INT.2 Layering
	ADV_LLD.2 Semiformal low-level design
	ADV_RCR.2 Semiformal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	**ALC_DVS.2 Sufficiency of security measures**
	ALC_LCD.2 Standardised life-cycle model
	ALC_TAT.3 Compliance with implementation standards - all parts
Tests	**ATE_COV.3 Ordered testing**
	ATE_DPT.3 Testing - low level design
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
Vulnerability assessment	**AVA_CCA.2 Systematic covert channel analysis**
	AVA_MSU.2 Misuse analysis - independent verification
	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.4 Highly resistant

Table 4.7 -EAL6

4.2.7 Evaluation assurance level 7 (EAL7) - formally verified design and tested

Objectives

178 EAL7 represents an achievable upper bound on evaluation assurance for practically useful products and should only be considered for experimental application to all but conceptually simple and well understood products.

179 EAL7 is therefore applicable to the development of specialist security products for application in extremely high risk situations and/or where the high value of the assets justifies the higher costs. Practical application of EAL7 is currently limited to products with tightly focused security functionality which is amenable to formal analysis.

Assurance components

180 **EAL7** (see Table **4.8**) provides assurance by an analysis of the security functions using a functional and interface specification, the high-level design of the subsystems, the low-level design of the modules of the TOE, and a structured presentation of the implementation, to understand the security behaviour. Assurance is additionally gained through a formal model**, a formal presentation of the functional specification and high-level design**, a semiformal presentation of the low-level design, and **formal and** semiformal demonstration of correspondence between them**, as appropriate**.

181 The analysis is supported by independent testing of the security functions, evidence of developer "**white** box" testing, **complete** independent confirmation of the developer test results, evidence of a developer search for obvious vulnerabilities (e.g., those in the public domain), and an independent search for vulnerabilities ensuring high resistance to penetration attack. The analysis also includes a systematic search for covert channels, when applicable, and is supported by requiring a modular, layered, **and simple** TOE design.

182 **EAL7** also provides assurance through the use of a structured development process, development environment controls, and comprehensive TOE configuration management including complete automation.

183 This EAL represents a meaningful increase in assurance from EAL6 by requiring more comprehensive analysis using formal representations and formal correspondence, comprehensive testing, and exhaustive covert channel analysis**.**

Assurance class	Assurance components
Configuration management	ACM_AUT.2 Complete CM automation
	ACM_CAP.4 Advanced support
	ACM_SCP.3 Development tools CM coverage
Delivery and operation	ADO_IGS.1 Installation, generation, and start-up procedures
Development	**ADV_FSP.6 Formal specification of the TSF properties**
	ADV_HLD.5 Formal high-level design
	ADV_IMP.3 Structured implementation of the TSF
	ADV_INT.3 Minimisation of Complexity
	ADV_LLD.2 Semiformal low-level design
	ADV_RCR.3 Formal correspondence demonstration
Guidance documents	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	ALC_DVS.2 Sufficiency of security measures
	ALC_LCD.3 Measurable life-cycle model
	ALC_TAT.3 Compliance with implementation standards - all parts
Tests	ATE_COV.3 Ordered testing
	ATE_DPT.4 Testing - implementation
	ATE_FUN.1 Functional testing
	ATE_IND.3 Independent testing - complete
Vulnerability assessment	AVA_CCA.2 Systematic covert channel analysis
	AVA_MSU.2 Misuse analysis - independent verification
	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.4 Highly resistant

Table 4.8 -EAL7

Chapter 5

Assurance classes, families, and components

184 This chapter provides the detailed requirements, presented in alphabetical order, of each of the assurance components, grouped by class and family.

Class ACM

Configuration management

185 Configuration management (CM) is an aspect of establishing that the functional requirements and specifications are realised in the implementation of the TOE. CM meets these objectives by requiring discipline and control in the processes of refinement and modification of the TOE. CM systems are put in place to ensure the integrity of the configuration items that they control, by providing a method of tracking these configuration items, and by ensuring that only authorised users are capable of changing them.

186 Figure 5.1 shows the families within this class, and the hierarchy of components within the families.

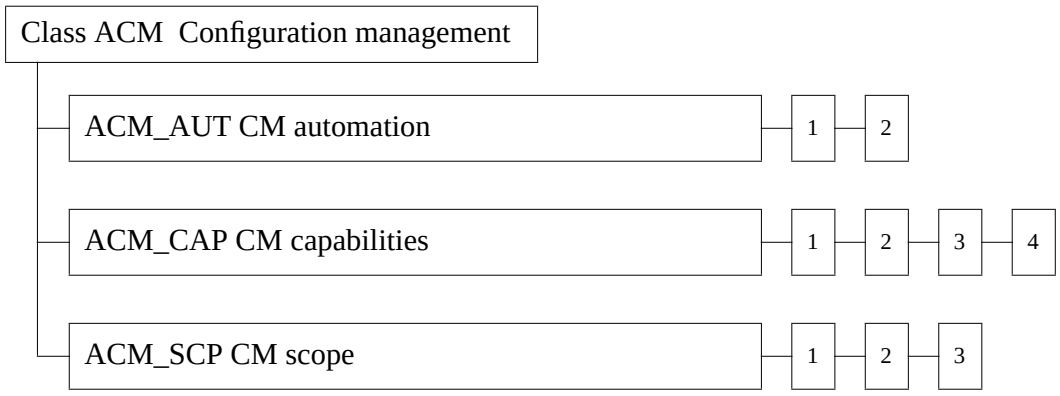

Figure 5.1 -Configuration management class decomposition

ACM_AUT CM automation

Objectives

187 The objective of introducing automated CM tools is to increase the efficiency of the CM system, by simultaneously increasing the reliability of the CM system and reducing the cost of operating it. While both automated and manual CM systems can be bypassed, ignored, or insufficient to prevent unauthorised modification, automated systems are less susceptible to human error or negligence. In addition, while a manual CM system can accomplish all of the same things that an automated system can, manual systems are typically more costly to operate on an ongoing basis.

Component levelling

188 The components in this family are levelled on the basis of the set of configuration items which are controlled through automated means.

Application notes

189 For ACM_AUT.1 and ACM_AUT.2, there is a requirement that the automated CM system control changes to the implementation representation of the TOE. The TOE implementation representation refers to all hardware, software, and firmware that comprise the physical TOE. In the case of a software-only TOE, the implementation representation may consist solely of source and object code, but in other TOEs the implementation representation may refer to a combination of software, hardware, and firmware.

ACM_AUT.1 Partial CM automation

Objectives

190 In development environments where the implementation representation is complex or is being developed by multiple developers, it is difficult to control changes without the support of automated tools. In particular, these automated tools need to be able to support the numerous changes that occur during development and ensure that those changes are performed by authorised developers before their application. It is the objective of this component to ensure that the implementation representation is controlled through automated means.

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ACM_AUT.1.1D **The developer shall provide a CM plan.**

Content and presentation of evidence elements:

ACM_AUT.1.1C **The CM plan shall describe the automated tools used in the CM system.**

ACM_AUT.1.2C **The CM plan shall describe how the automated tools are used in the CM system.**

ACM_AUT.1.3C **The CM system shall provide an automated means to ensure that only authorised changes are made to the TOE implementation representation.**

ACM_AUT.1.4C **The CM system shall provide an automated means to support the generation of any supported TSF from its implementation representation.**

ACM_AUT.1.5C **The CM system shall provide an automated means to support the comparison of any two supported TSF versions, to ascertain the changes.**

Evaluator action elements:

ACM_AUT.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ACM_AUT.2 Complete CM automation

Objectives

191 In development environments where the configuration items are complex or are being developed by multiple developers, it is difficult to control changes without the support of automated tools. In particular, these automated tools need to be able to support the numerous changes that occur during development and ensure that those changes are performed by authorised developers before their application. It is the objective of this component to ensure that all configuration items are controlled through automated means.

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ACM_AUT.2.1D The developer shall provide a CM plan.

Content and presentation of evidence elements:

ACM_AUT.2.1C The CM plan shall describe the automated tools used in the CM system.

ACM_AUT.2.2C The CM plan shall describe how the automated tools are used in the CM system.

ACM_AUT.2.3C The CM system shall provide an automated means to ensure that only authorised changes are made to the TOE implementation representation**, and to all other configuration items.**

ACM_AUT.2.4C The CM system shall provide an automated means to support the generation of any supported TSF from its implementation representation.

ACM_AUT.2.5C The CM system shall provide an automated means to support the comparison of any two supported TSF versions, to ascertain the changes.

ACM_AUT.2.6C **The CM system shall provide an automated means to identify all other configuration items that are affected by the modification of a given configuration item.**

Evaluator action elements:

ACM_AUT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_CAP CM capabilities

Objectives

192 The capabilities of the CM system address the likelihood that accidental or unauthorised modifications of the configuration items will occur. The CM system should ensure the integrity of the TSF from the early design stages through all subsequent maintenance efforts.

193 The objectives of this family include the following:

a) ensuring that the TSF is correct and complete before it is sent to the consumer;

b) ensuring that no configuration items are missed during evaluation;

c) preventing unauthorised modification, addition, or deletion of TOE configuration items; and

d) enabling recovery to an earlier version of the TOE, in the event that an error occurs through modification, addition, or deletion of TOE configuration items.

Component levelling

194 The components in this family are levelled on the basis of what the CM system's capabilities are, the scope of the CM documentation provided by the developer, and whether the developer provides justification that the CM system meets its security requirements.

Application notes

195 For ACM_CAP.1 and the higher components, there is a requirement that a configuration list be provided. The configuration list contains all configuration items which are maintained by the CM system.

196 For ACM_CAP.2 and the higher components, there is a requirement that the CM documentation include evidence that the CM system is working properly. An example of such evidence might be audit trail output from the CM system. The evaluator is responsible for examining such evidence, to determine that it is sufficient to demonstrate proper functionality of the CM system.

197 For ACM_CAP.2 and the higher components, there is a requirement that evidence be provided that all configuration items are being maintained under the CM system. Since a configuration item refers to an item which is on the configuration list, this requirement states that all items on the configuration list are maintained under the CM system.

198 For ACM_CAP.3 and ACM_CAP.4, there is a requirement that the CM system support the generation of all supported versions of the TOE. This provides the

ability to recover to a previous known version in the event that an error occurs through modification, addition or deletion of TOE configuration items.

ACM_CAP.1 Minimal support

Objectives

199 Clear identification of the TOE is required to determine those items under evaluation that are subject to the criteria requirements.

Dependencies:

No dependencies.

Developer action elements:

ACM_CAP.1.1D **The developer shall use a CM system.**

ACM_CAP.1.2D **The developer shall provide CM documentation.**

Content and presentation of evidence elements:

ACM_CAP.1.1C **The CM documentation shall include a configuration list.**

ACM_CAP.1.2C **The configuration list shall describe the configuration items that comprise the TOE.**

ACM_CAP.1.3C **The CM documentation shall describe the method used to uniquely identify the TOE configuration items.**

Evaluator action elements:

ACM_CAP.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ACM_CAP.2 Authorisation controls

Objectives

200 Clear identification of the TOE is required to determine those items under evaluation that are subject to the criteria requirements.

201 Assurance of TOE integrity may be gained by controlling the ability to modify the TOE configuration items. Ensuring proper functionality and use of the CM system also provides assurance that the CM system is correctly enforcing the integrity of the TOE.

Dependencies:

ACM_SCP.1 Minimal CM coverage

ALC_DVS.1 Identification of security measures

Developer action elements:

ACM_CAP.2.1D The developer shall use a CM system.

ACM_CAP.2.2D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_CAP.2.1C The CM documentation shall include a configuration list **and a CM plan**.

ACM_CAP.2.2C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.2.3C The CM documentation shall describe the method used to uniquely identify the TOE configuration items.

ACM_CAP.2.4C **The CM plan shall describe how the CM system is used.**

ACM_CAP.2.5C **The CM documentation shall provide evidence that the CM system is working properly.**

ACM_CAP.2.6C **The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.**

ACM_CAP.2.7C **The CM system shall ensure that only authorised changes are made to the TOE configuration items.**

Evaluator action elements:

ACM_CAP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_CAP.3 Generation support and acceptance procedures

Objectives

202 Clear identification of the TOE is required to determine those items under evaluation that are subject to the criteria requirements.

203 Assurance of TOE integrity may be gained by controlling the ability to modify the TOE configuration items. Ensuring proper functionality and use of the CM system also provides assurance that the CM system is correctly enforcing the integrity of the TOE.

204 The ability to generate previous but still supported versions of the TOE is necessary for the resolution of any new flaws discovered during operation.

205 The purpose of acceptance procedures is to confirm that any creation or modification of TSF configuration items is authorised.

Dependencies:

ACM_SCP.1 Minimal CM coverage

ALC_DVS.1 Identification of security measures

Developer action elements:

ACM_CAP.3.1D The developer shall use a CM system.

ACM_CAP.3.2D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_CAP.3.1C The CM documentation shall include a configuration list**,** a CM plan**, and an acceptance plan.**

ACM_CAP.3.2C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.3.3C The CM documentation shall describe the method used to uniquely identify the TOE configuration items.

ACM_CAP.3.4C The CM plan shall describe how the CM system is used.

ACM_CAP.3.5C The CM documentation shall provide evidence that the CM system is working properly.

ACM_CAP.3.6C The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.

ACM_CAP.3.7C The CM system shall ensure that only authorised changes are made to the TOE configuration items.

ACM_CAP.3.8C **The CM system shall support the generation of all supported versions of the TOE.**

ACM_CAP.3.9C **The acceptance plan shall describe the procedures used to accept modified or newly created TSF configuration items as part of the TOE.**

Evaluator action elements:

ACM_CAP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_CAP.4 Advanced support

Objectives

206 Clear identification of the TOE is required to determine those items under evaluation that are subject to the criteria requirements.

207 Assurance of TOE integrity may be gained by controlling the ability to modify the TOE configuration items. Ensuring proper functionality and use of the CM system also provides assurance that the CM system is correctly enforcing the integrity of the TOE.

208 The ability to generate previous but still supported versions of the TOE is necessary for the resolution of any new flaws discovered during operation.

209 The purpose of acceptance procedures is to confirm that any creation or modification of TSF configuration items is authorised.

210 Integration procedures ensure that the introduction of modifications into the TSF is performed in a controlled and complete manner.

211 Requiring that the CM system be able to identify the master copy of the material used to generate the TSF helps to ensure that the integrity of this material is preserved by the appropriate technical, physical and procedural safeguards.

Dependencies:

ACM_SCP.1 Minimal CM coverage

ALC_DVS.2 Sufficiency of security measures

Developer action elements:

ACM_CAP.4.1D The developer shall use a CM system.

ACM_CAP.4.2D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_CAP.4.1C The CM documentation shall include a configuration list, a CM plan, an acceptance plan**, and integration procedures.**

ACM_CAP.4.2C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.4.3C The CM documentation shall describe the method used to uniquely identify the TOE configuration items.

ACM_CAP.4.4C The CM plan shall describe how the CM system is used.

ACM_CAP.4.5C The CM documentation shall provide evidence that the CM system is working properly.

ACM_CAP.4.6C The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.

ACM_CAP.4.7C The CM system shall ensure that only authorised changes are made to the TOE configuration items.

ACM_CAP.4.8C The CM system shall support the generation of all supported versions of the TOE.

ACM_CAP.4.9C The acceptance plan shall describe the procedures used to accept modified or newly created TSF configuration items as part of the TOE.

ACM_CAP.4.10C **The integration procedures shall describe how the CM system is applied in the TOE manufacturing process.**

ACM_CAP.4.11C **The CM system shall require that the person responsible for accepting a configuration item into CM is not the person who developed it.**

ACM_CAP.4.12C **The CM system shall permit clear identification of the TSF.**

ACM_CAP.4.13C **The CM system shall support the audit of all modifications to the TSF, including as a minimum the originator, date, and time in the audit trail.**

ACM_CAP.4.14C **The CM system shall be able to identify the master copy of all material used to generate the TSF.**

ACM_CAP.4.15C **The evidence shall justify that the use of the CM system is sufficient to ensure that only authorised changes are made to the TOE.**

ACM_CAP.4.16C **The evidence shall justify that the integration procedures ensure that the introduction of modifications into the TSF is performed in a controlled and complete manner.**

ACM_CAP.4.17C **The evidence shall justify that the CM system is sufficient to ensure that the person responsible for accepting a configuration item into CM is not the person who developed it.**

ACM_CAP.4.18C **The evidence shall justify that the acceptance procedures provide for an adequate and appropriate review of changes to TSF configuration items.**

Evaluator action elements:

ACM_CAP.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_SCP CM scope

Objectives

212 The objective is to ensure that all necessary TOE configuration items are tracked by the CM system. This helps to ensure that the integrity of these configuration items is protected through the capabilities of the CM system.

213 The objectives of this family include the following:

a) ensuring that the TOE implementation representation is tracked;

b) ensuring that all necessary documentation, including problem reports, are tracked during development and operation;

c) ensuring that configuration options (e.g. compiler switches) are tracked; and

d) ensuring that development tools are tracked.

Component levelling

214 The components in this family are levelled on the basis of which of the following are tracked by the CM system: the TOE implementation representation, design documentation; test documentation; user documentation; administrator documentation; CM documentation; security flaws; and development tools.

Application notes

215 For ACM_SCP.1 and the higher components, there is a requirement that the TOE implementation representation be tracked by the CM system. The TOE implementation representation refers to all hardware, software, and firmware that comprise the physical TOE. In the case of a software-only TOE, the implementation representation may consist solely of source and object code, but in other TOEs the implementation representation may refer to a combination of software, hardware, and firmware.

216 For ACM_SCP.2 and ACM_SCP.3, there is a requirement that security flaws be tracked by the CM system. This requires that information regarding previous security flaws and their resolution be maintained, as well as details regarding current security flaws.

217 For ACM_SCP.3, there is a requirement that development tools and other related information be tracked by the CM system. Examples of development tools are programming languages and compilers. Information pertaining to TOE generation items (such as compiler options, installation/generation options, and build options) is an example of information relating to development tools.

ACM_SCP.1 Minimal CM coverage

Objectives

218 A CM system can control changes only to those items that have been placed under CM. At a minimum, the TOE implementation representation, design, tests, user and administrator documentation, and CM documentation should be placed under CM.

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ACM_SCP.1.1D **The developer shall provide CM documentation.**

Content and presentation of evidence elements:

ACM_SCP.1.1C **As a minimum, the following shall be tracked by the CM system: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, and CM documentation.**

ACM_SCP.1.2C **The CM documentation shall describe how configuration items are tracked by the CM system.**

Evaluator action elements:

ACM_SCP.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ACM_SCP.2 Problem tracking CM coverage

Objectives

219 A CM system can control changes only to those items that have been placed under CM. At a minimum, the TOE implementation representation, design, tests, user and administrator documentation, and CM documentation should be placed under CM.

220 The ability to track security flaws under CM ensures that security flaw reports are not lost or forgotten, and allows a developer to track security flaws to their resolution.

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ACM_SCP.2.1D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_SCP.2.1C As a minimum, the following shall be tracked by the CM system: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, CM documentation**, and security flaws.**

ACM_SCP.2.2C The CM documentation shall describe how configuration items are tracked by the CM system.

Evaluator action elements:

ACM_SCP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACM_SCP.3 Development tools CM coverage

Objectives

221 A CM system can control changes only to those items that have been placed under CM. At a minimum, the TOE implementation representation, design, tests, user and administrator documentation, and CM documentation should be placed under CM.

222 The ability to track security flaws under CM ensures that security flaw reports are not lost or forgotten, and allows a developer to track security flaws to their resolution.

223 Development tools play an important role in ensuring the production of a quality version of the TSF. Therefore, it is important to control modifications to these tools.

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ACM_SCP.3.1D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_SCP.3.1C As a minimum, the following shall be tracked by the CM system: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, CM documentation, security flaws**, and development tools and related information.**

ACM_SCP.3.2C The CM documentation shall describe how configuration items are tracked by the CM system.

Evaluator action elements:

ACM_SCP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Class ADO

Delivery and operation

224 Delivery and operation provides requirements for correct delivery, installation, generation, and start-up of the TOE.

225 Figure 5.2 shows the families within this class, and the hierarchy of components within the families.

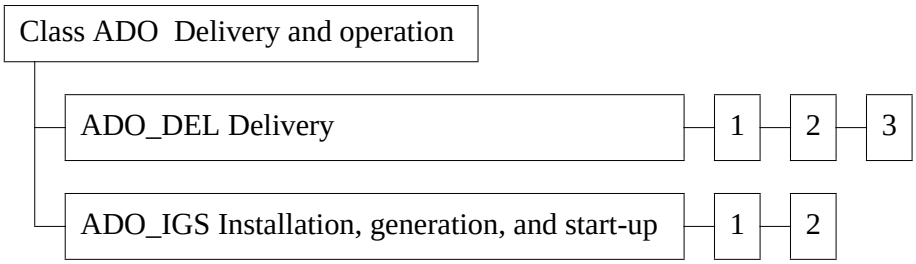

Figure 5.2 -Delivery and operation class decomposition

ADO_DEL Delivery

Objectives

226 The requirements for delivery call for system control and distribution facilities and procedures that provide assurance that the recipient receives the TOE that the sender intended to send, without any modifications. For a valid delivery, what is received must correspond precisely to the TOE master copy, thus avoiding any tampering with the actual version, or substitution of a false version.

Component levelling

227 The components in this family are levelled on the basis of increasing requirements on the developer to detect and prevent modifications to the TOE during delivery.

Application notes

228 The PP/ST author should consider whether it would be useful to introduce the assurance provided by a delivery component into the PP/ST. This should receive special attention as no delivery component is included in any EAL and the absence of such components decreases the assurance that the TOE received is the same as the one that was evaluated. In considering which delivery component to select, the selected EAL and intended application of the TOE should be primary factors. In general, higher ADO_DEL components are more appropriate for the higher EALs and for very sensitive applications.

ADO_DEL.1 Delivery procedures

Dependencies:

No dependencies.

Developer action elements:

ADO_DEL.1.1D **The developer shall provide documentation about the procedures for delivery of the TOE or parts of it to the user.**

ADO_DEL.1.2D **The developer shall use the delivery procedures.**

Content and presentation of evidence elements:

ADO_DEL.1.1C **The delivery documentation shall describe the procedures to be employed when distributing versions of the TOE to a user's site.**

Evaluator action elements:

ADO_DEL.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADO_DEL.2 Detection of modification

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ADO_DEL.2.1D The developer shall provide documentation about the procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.2.2D The developer shall use the delivery procedures.

Content and presentation of evidence elements:

ADO_DEL.2.1C The delivery documentation shall describe the procedures to be employed when distributing versions of the TOE to a user's site.

ADO_DEL.2.2C **The delivery documentation shall state how the procedures are to be employed to detect modifications.**

ADO_DEL.2.3C **The delivery documentation shall describe how the various procedures and technical measures provide for the detection of modifications, or any discrepancy between the developer's master copy and the version received at the user site.**

ADO_DEL.2.4C **The delivery documentation shall describe how the various procedures allow detection of attempted masquerading even in cases in which the developer has sent nothing to the user's site.**

Evaluator action elements:

ADO_DEL.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_DEL.3 Prevention of modification

Dependencies:

ACM_CAP.2 Authorisation controls

Developer action elements:

ADO_DEL.3.1D The developer shall provide documentation about the procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.3.2D The developer shall use the delivery procedures.

Content and presentation of evidence elements:

ADO_DEL.3.1C The delivery documentation shall describe the procedures to be employed when distributing versions of the TOE to a user's site.

ADO_DEL.3.2C The delivery documentation shall state how the procedures are to be employed to detect modification.

ADO_DEL.3.3C The delivery documentation shall describe how the various procedures and technical measures provide for the **prevention** of modifications, or any discrepancy between the developer's master copy and the version received at the user site.

ADO_DEL.3.4C The delivery documentation shall describe how the various procedures allow detection of attempted masquerading even in cases in which the developer has sent nothing to the user's site.

Evaluator action elements:

ADO_DEL.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS Installation, generation, and start-up

Objectives

229 Installation, generation, and start-up procedures are useful for ensuring that the TOE has been installed, generated, and started in a secure manner as intended by the developer.

Component levelling

230 The components in this family are levelled on the basis of whether the TOE generation options are audited.

Application notes

231 The generation requirements are applicable only to TOEs that provide the ability to generate an operational TOE from source or object code.

232 The installation, generation, and start-up procedures may exist as a separate document, but would typically be grouped with other administrative guidance.

ADO_IGS.1 Installation, generation, and start-up procedures

Dependencies:

AGD_ADM.1 Administrator guidance

Developer action elements:

ADO_IGS.1.1D **The developer shall document procedures to be used for the secure installation, generation, and start-up of the TOE.**

Content and presentation of evidence elements:

ADO_IGS.1.1C **The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.**

Evaluator action elements:

ADO_IGS.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADO_IGS.2 Generation log

Dependencies:

AGD_ADM.1 Administrator guidance

Developer action elements:

ADO_IGS.2.1D The developer shall document procedures to be used for the secure installation, generation, and start-up of the TOE.

Content and presentation of evidence elements:

ADO_IGS.2.1C The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

ADO_IGS.2.2C **The documentation shall describe procedures capable of creating a log containing the generation options used to generate the TOE in such a way that it is possible to determine exactly how and when the TOE was generated.**

Evaluator action elements:

ADO_IGS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence**.**

Class ADV

Development

233 The development class encompasses four families of requirements for representing the TSF at various levels of abstraction from the functional interface to the implementation. The development class also includes a family of requirements for a correspondence mapping between the various TSF representations, ultimately requiring a demonstration of correspondence from the least abstract representation through all intervening representations to the TOE summary specification provided in the ST. The other family in the development class describes requirements for the internal structure of the TSF.

234 Figure 5.3 shows the families within this class, and the hierarchy of components within the families.

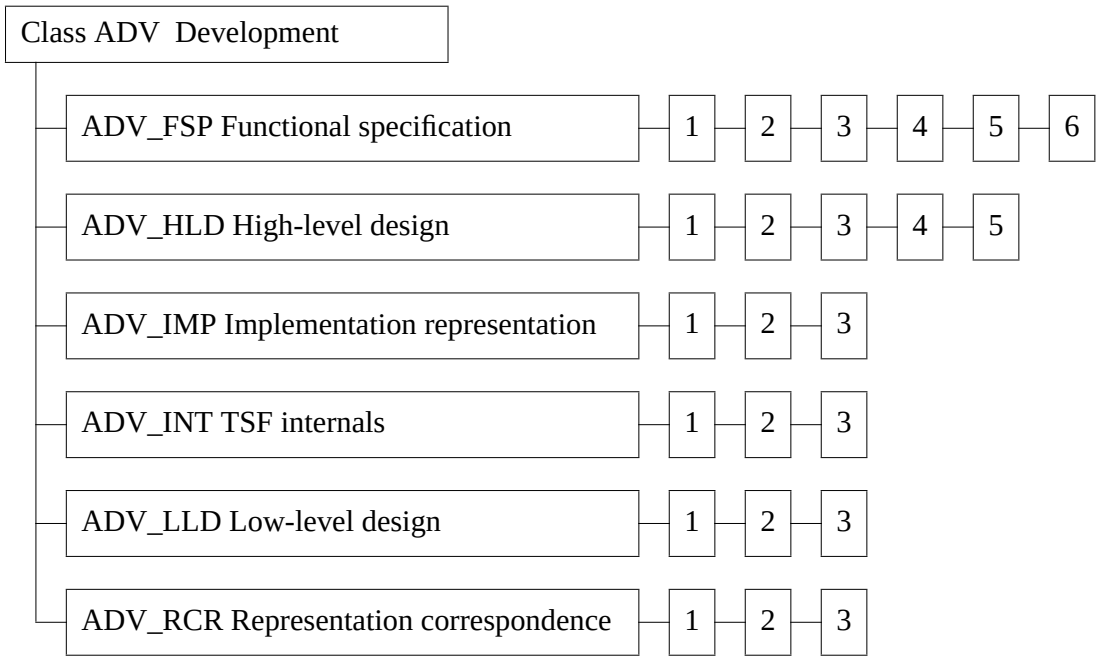

Figure 5.3 - Development class decomposition

235 The paradigm evident for these families is one of a functional specification of the TSF, decomposing the TSF into subsystems, decomposing the subsystems into modules, showing the implementation of the modules, and demonstration of correspondence between all decompositions that are provided as evidence. The requirements for the various TSF representations are separated into different families, however, since some of the representations are not necessary for low assurance evaluations.

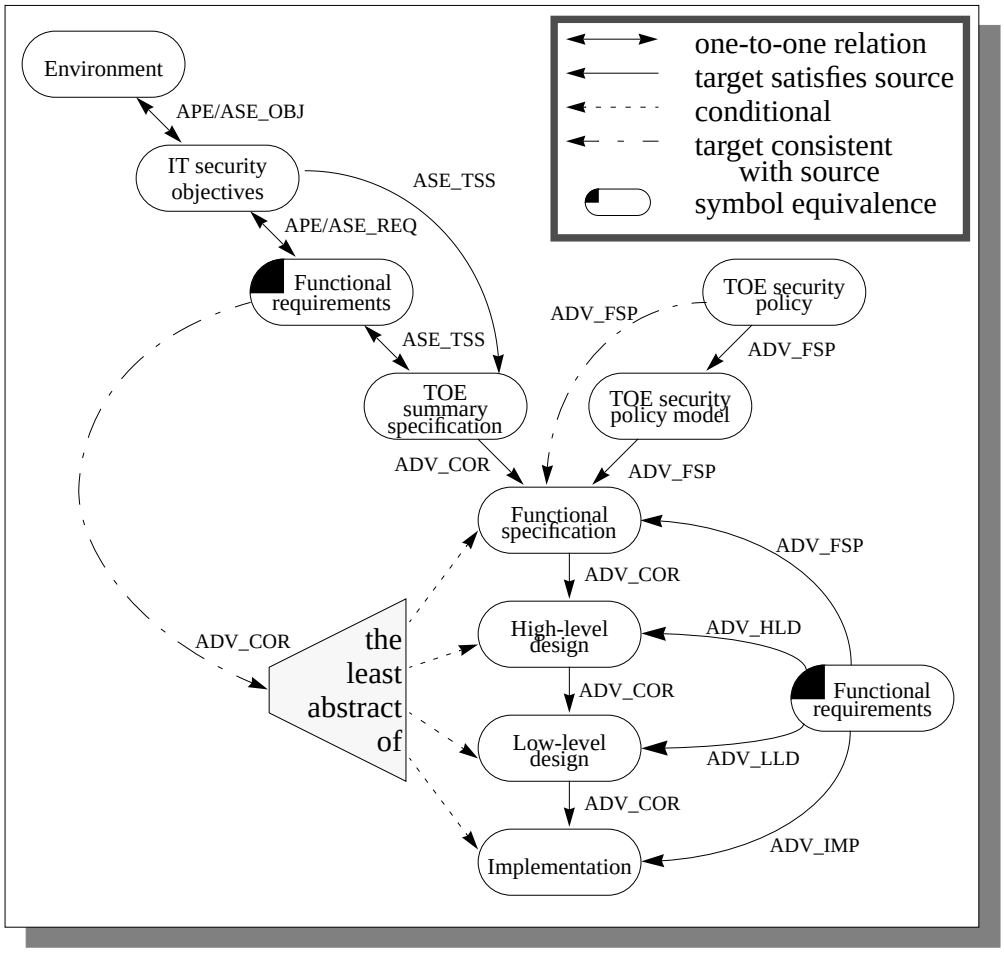

Figure 5.4 - Relationships between TOE representations and requirements

236 Figure 5.4 indicates the relationships between the various TSF representations and the objectives and requirements that they are intended to address. As the figure indicates, the Protection Profile evaluation (APE) and/or the Security Target evaluation (ASE) classes define the requirements for the correspondence between the functional requirements and the IT security objectives as well as between the IT security objectives and the TOE's anticipated environment. Class ASE also defines requirements for the correspondence between both the IT security objectives and functional requirements and the TOE summary specification.

237 The requirements for all other correspondence shown in Figure 5.4 are defined in the ADV class. The ADV_FSP family defines the requirements for: correspondence between the TSP and TSP model; correspondence between the TSP model and functional specification; and, consistency between the TSP and functional specification. The ADV_RCR family defines the requirements for correspondence between all available TSF representations from the TOE summary specification through the implementation representation and further defines

requirements for consistency between the most detailed (or least abstract) TSF representation provided and the functional requirements. Finally, each assurance family specific to a TSF representation (e.g., ADV_FSP - functional specification) defines requirements relating the TSF representation to the functional requirements, the combination of which will ensure that the functional requirements have all been addressed.

ADV_FSP Functional specification

Objectives

238 The functional specification is a high-level description of the user-visible interface and behaviour of the TSF. It is a refinement of the statement of IT functional requirements in the ST of the TOE. The functional specification has to show that all the functional requirements defined in the ST are addressed, and that the TSP is enforced by the TSF.

Component levelling

239 The components in this family are levelled on the basis of the degree of formalism required of the functional specification and any TSP model, and according to the degree of rigour that is required to demonstrate that the functional specification is consistent with the TSP model.

Application notes

240 In addition to the content indicated in the following requirements, the functional specification shall also include any additional specific detail specified by the documentation notes in the related functional components.

241 The developer must provide evidence that the TSF is completely represented by the functional specification. While a functional specification for the entire TOE would allow an evaluator to determine the TSF boundary, it is not necessary to require that specification when other evidence could be provided to demonstrate the TSF boundary.

242 The evaluator of the TOE is expected to make determinations regarding the functional requirements in the ST relevant to the functional specification. In the course of the functional specification evaluation there are essentially three types of evaluator determination: specific functional requirements are met and no further work (e.g., with a less abstract representation of the TSF) is necessary; specific functional requirements are violated and the TOE fails to meet its requirements; and specific functional requirements have not been addressed and further analysis (of another TSF representation) is necessary. Whenever more analysis is necessary, the evaluator is expected to carry that information forward to the analysis of other TSF representations. If requirements are not addressed after the analysis of the last provided TSF representation, this also represents a failure of the TOE evaluation. Note that this more comprehensive failure determination requirement is realised in the Representation correspondence (ADV_RCR) family.

243 In all cases, it is important that the evaluator evaluate the TSF as a unit since in many cases the security functions must cooperate to meet specific functional requirements and also each security function must not interfere with the operation of any other security function.

244 While a TSP may represent any policies, TSP models have traditionally represented only subsets of those policies. As a result, the TSP model cannot be treated like

every other TSF representation inasmuch as the correspondence between the TSP model to the adjacent abstractions (i.e., TSP and functional specification) may not be complete. As a result, there must be a demonstration of correspondence from the functional specification to the TSP directly, rather than through the intervening representation (i.e., TSP model) where correspondence may be lost. For these reasons, all of the requirements for correspondence between the TSP, TSP model, and functional specification have been included in this family and the correspondence requirements in the Representation correspondence (ADV_RCR) family do not apply to the TSP and TSP model.

245 Beginning with ADV_FSP.1, requirements are defined to ensure that the functional specification is consistent with the TSP. Beginning with ADV_FSP.2, because there is no requirement for a TSP model in ADV_FSP.1, requirements are defined to describe the rules and characteristics of applicable policies of the TSP in the TSP model and to ensure that the TSP model satisfies the corresponding policies of the TSP. The “rules” and “characteristics” of a TSP model are intended to allow flexibility in the type of model that may be developed (e.g., state transition, non-interference). For example, rules may be represented as “properties” (e.g., simple security property) and characteristics may be represented as definitions such as “initial state”, “secure state”, “subjects”, and “objects”.

246 Since not all policies can be modeled, given the current state of the art, the requirement indicating which policies shall be modeled is subjective. The PP/ST author should identify specific functions and associated policies that are required to be modeled. At the very least, access control policies are expected to be modeled since they are currently within the state of the art.

ADV_FSP.1 TOE and security policy

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.1.1D **The developer shall provide a functional specification.**

ADV_FSP.1.2D **The developer shall provide a TSP.**

Content and presentation of evidence elements:

ADV_FSP.1.1C **The functional specification shall describe the TSF using an informal style.**

ADV_FSP.1.2C **The functional specification shall include an informal presentation of syntax and semantics of all external TSF interfaces.**

ADV_FSP.1.3C **The functional specification shall include evidence that demonstrates that the TSF is completely represented.**

Evaluator action elements:

ADV_FSP.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADV_FSP.1.2E **The evaluator shall determine that the functional specification is consistent with the TSP.**

ADV_FSP.1.3E **The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.**

ADV_FSP.2 Informal security policy model

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.2.1D The developer shall provide a functional specification.

ADV_FSP.2.2D The developer shall provide a TSP.

ADV_FSP.2.3D **The developer shall provide an informal TSP model.**

ADV_FSP.2.4D **The developer shall provide a demonstration of correspondence between the informal TSP model and the functional specification.**

Content and presentation of evidence elements:

ADV_FSP.2.1C The functional specification shall describe the TSF using an informal style.

ADV_FSP.2.2C The functional specification shall include an informal presentation of syntax and semantics of all external TSF interfaces.

ADV_FSP.2.3C The functional specification shall include evidence that demonstrates that the TSF is completely represented.

ADV_FSP.2.4C **The demonstration of correspondence between the informal TSP model and the functional specification shall describe how the functional specification satisfies the informal TSP model.**

ADV_FSP.2.5C **The demonstration of correspondence between the informal TSP model and the functional specification shall show that there are no security functions in the functional specification that conflict with the informal TSP model.**

ADV_FSP.2.6C **The informal TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.**

ADV_FSP.2.7C **The informal TSP model shall include a rationale that demonstrates that policies of the TSP that are modeled are satisfied by the informal TSP model.**

ADV_FSP.2.8C **The informal TSP model shall justify that all policies of the TSP that can be modeled are represented in the informal TSP model.**

Evaluator action elements:

ADV_FSP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.2.2E The evaluator shall determine that the functional specification is consistent with the TSP.

ADV_FSP.2.3E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_FSP.3 Semiformal security policy model

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.3.1D The developer shall provide a functional specification.

ADV_FSP.3.2D The developer shall provide a TSP.

ADV_FSP.3.3D The developer shall provide **a semiformal** TSP model.

ADV_FSP.3.4D The developer shall provide a demonstration of correspondence between the **semiformal** TSP model and the functional specification.

Content and presentation of evidence elements:

ADV_FSP.3.1C The functional specification shall describe the TSF using an informal style.

ADV_FSP.3.2C The functional specification shall include an informal presentation of syntax and semantics of all external TSF interfaces.

ADV_FSP.3.3C The functional specification shall include evidence that demonstrates that the TSF is completely represented.

ADV_FSP.3.4C The demonstration of correspondence between the **semiformal** TSP model and the functional specification shall describe how the functional specification satisfies the **semiformal** TSP model.

ADV_FSP.3.5C The demonstration of correspondence between the **semiformal** TSP model and the functional specification shall show that there are no security functions in the functional specification that conflict with the **semiformal** TSP model.

ADV_FSP.3.6C The **semiformal** TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_FSP.3.7C The **semiformal** TSP model shall include a rationale that demonstrates that policies of the TSP that are modeled are satisfied by the **semiformal** TSP model.

ADV_FSP.3.8C The **semiformal** TSP model shall justify that all policies of the TSP that can be modeled are represented in the **semiformal** TSP model.

Evaluator action elements:

ADV_FSP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.3.2E The evaluator shall determine that the functional specification is consistent with the TSP.

ADV_FSP.3.3E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_FSP.4 Formal security policy model

Application notes

247 The requirement for both an informal and semiformal functional specification is necessary to allow an evaluator to effectively comprehend and evaluate the semiformal representation using the informal representation for support.

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.4.1D The developer shall provide a functional specification.

ADV_FSP.4.2D The developer shall provide a TSP.

ADV_FSP.4.3D The developer shall provide a **formal** TSP model.

ADV_FSP.4.4D The developer shall provide a demonstration of correspondence between the **formal** TSP model and the functional specification.

Content and presentation of evidence elements:

ADV_FSP.4.1C The functional specification shall describe the TSF using **both** an informal **and semiformal** style.

ADV_FSP.4.2C The functional specification shall include **both** an informal **and semiformal** presentation of syntax**, effects, exceptions, error messages,** and semantics of all external TSF interfaces.

ADV_FSP.4.3C The functional specification shall include evidence that demonstrates that the TSF is completely represented.

ADV_FSP.4.4C The demonstration of correspondence between the **formal** TSP model and the functional specification shall describe how the functional specification satisfies the **formal** TSP model.

ADV_FSP.4.5C The demonstration of correspondence between the **formal** TSP model and the functional specification shall show that there are no security functions in the functional specification that conflict with the **formal** TSP model.

ADV_FSP.4.6C The **formal** TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_FSP.4.7C The **formal** TSP model shall include a rationale that demonstrates that policies of the TSP that are modeled are satisfied by the **formal** TSP model.

ADV_FSP.4.8C The **formal** TSP model shall justify that all policies of the TSP that can be modeled are represented in the **formal** TSP model.

Evaluator action elements:

ADV_FSP.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.4.2E The evaluator shall determine that the functional specification is consistent with the TSP.

ADV_FSP.4.3E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_FSP.5 Property specification by model interpretation

Application notes

248 The requirement for both an informal and semiformal functional specification is necessary to allow an evaluator to effectively comprehend and evaluate the semiformal representation using the informal representation for support.

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.5.1D The developer shall provide a functional specification.

ADV_FSP.5.2D The developer shall provide a TSP.

ADV_FSP.5.3D The developer shall provide a formal TSP model.

ADV_FSP.5.4D The developer shall provide a demonstration of correspondence between the formal TSP model and the functional specification.

Content and presentation of evidence elements:

ADV_FSP.5.1C The functional specification shall describe the TSF using both an informal and semiformal style.

ADV_FSP.5.2C The functional specification shall include both an informal and semiformal presentation of syntax, effects, exceptions, error messages, and semantics of all external TSF interfaces.

ADV_FSP.5.3C The functional specification shall include evidence that demonstrates that the TSF is completely represented.

ADV_FSP.5.4C The demonstration of correspondence between the formal TSP model and the functional specification shall describe how the functional specification satisfies the formal TSP model.

ADV_FSP.5.5C The demonstration of correspondence between the formal TSP model and the functional specification shall show that there are no security functions in the functional specification that conflict with the formal TSP model.

ADV_FSP.5.6C The formal TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_FSP.5.7C The formal TSP model shall include a rationale that demonstrates that policies of the TSP that are modeled are satisfied by the formal TSP model.

ADV_FSP.5.8C The formal TSP model shall justify that all policies of the TSP that can be modeled are represented in the formal TSP model.

ADV_FSP.5.9C **The evidence shall justify that the informal and semiformal functional specifications are consistent.**

Evaluator action elements:

ADV_FSP.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.5.2E The evaluator shall determine that the functional specification is consistent with the TSP.

ADV_FSP.5.3E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_FSP.6 Formal specification of the TSF properties

Application notes

249 The requirement for both an informal and formal functional specification is necessary to allow an evaluator to effectively comprehend and evaluate the more formal representation using the informal representation for support.

Dependencies:

ASE_TSS.1 Security Target, TOE Summary Specification, Evaluation Requirements

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_FSP.6.1D The developer shall provide a functional specification.

ADV_FSP.6.2D The developer shall provide a TSP.

ADV_FSP.6.3D The developer shall provide a formal TSP model.

ADV_FSP.6.4D The developer shall provide a **proof** of correspondence between the formal TSP model and the functional specification.

Content and presentation of evidence elements:

ADV_FSP.6.1C The functional specification shall describe the TSF using both an informal and **formal** style.

ADV_FSP.6.2C The functional specification shall include both an informal and **formal** presentation of syntax, effects, exceptions, error messages, and semantics of all external TSF interfaces.

ADV_FSP.6.3C The functional specification shall include evidence that demonstrates that the TSF is completely represented.

ADV_FSP.6.4C The **proof** of correspondence between the formal TSP model and the functional specification shall **demonstrate that** the functional specification satisfies the formal TSP model.

ADV_FSP.6.5C The **proof** of correspondence between the formal TSP model and the functional specification shall **demonstrate** that there are no security functions in the functional specification that conflict with the formal TSP model.

ADV_FSP.6.6C The formal TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_FSP.6.7C The formal TSP model shall include a rationale that demonstrates that policies of the TSP that are modeled are satisfied by the formal TSP model.

ADV_FSP.6.8C The formal TSP model shall justify that all policies of the TSP that can be modeled are represented in the formal TSP model.

ADV_FSP.6.9C The evidence shall justify that the informal and **formal** functional specifications are consistent.

Evaluator action elements:

ADV_FSP.6.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.6.2E The evaluator shall determine that the functional specification is consistent with the TSP.

ADV_FSP.6.3E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_HLD High-level design

Objectives

250 The high-level design of a TOE provides a description of the TSF in terms of major structural units (i.e., subsystems) and relates these units to the functions that they contain. The high-level design provides assurance that the TOE provides an architecture appropriate to implement the claimed functional requirements.

251 The high-level design refines the functional specification into subsystems. For each subsystem of the TSF, the high-level design describes its purpose and function and identifies the security functions enforced by the subsystem. The interrelationships of all subsystems are also defined in the high-level design. These interrelationships will be represented as external interfaces for data flow, control flow, etc., as appropriate.

Component levelling

252 The components in this family are levelled on the basis of the degree of formalism required of the high-level design.

Application notes

253 In addition to the content indicated in the following requirements, the high-level design shall also include any additional specific detail specified by the documentation notes in the related functional components.

254 The developer is expected to describe the design of the TSF in terms of subsystems. The term “subsystem” is used here to express the idea of decomposing the TSF into a relatively small number of parts. While the developer is not required to actually have “subsystems”, the developer is expected to represent a similar level of decomposition. For example, a design may be similarly decomposed using “layers”, “domains”, or “servers”.

255 The evaluator of the TOE is expected to make determinations regarding the functional requirements in the ST relevant to the high-level design. In the course of the high-level design evaluation there are essentially three types of evaluator determination: specific functional requirements are met and no further work (e.g., with a less abstract representation of the TSF) is necessary; specific functional requirements are violated and the TOE fails to meet its requirements; and specific functional requirements have not been addressed and further analysis (of another TSF representation) is necessary. Whenever more analysis is necessary, the evaluator is expected to carry that information forward to the analysis of other TSF representations. If requirements are not addressed after the analysis of the last provided TSF representation, this also represents a failure of the TOE evaluation. Note that this more comprehensive failure determination requirement is realised in the Representation correspondence (ADV_RCR) family.

256 In all cases, it is important that the evaluator evaluate the TSF as a unit since in many cases the security functions must cooperate to meet specific functional

requirements and also each security function must not interfere with the operation of any other security function.

257 The term “security functionality” is used to represent operations that a subsystem performs that have some effect on the security functions implemented by the TOE. This distinction is made because design constructs, such as subsystems and modules, do not necessarily relate to specific security functions. While a given subsystem may correspond directly to a security function, or even multiple security functions, it is also possible that many subsystems must be combined to implement a single security function.

258 The term “TSP enforcing subsystems” refers to a subsystem that contributes to the enforcement of the TSP.

ADV_HLD.1 Descriptive high-level design

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_HLD.1.1D **The developer shall provide the high-level design of the TSF.**

Content and presentation of evidence elements:

ADV_HLD.1.1C **The presentation of the high-level design shall be informal.**

ADV_HLD.1.2C **The high-level design shall describe the structure of the TSF in terms of subsystems.**

ADV_HLD.1.3C **The high-level design shall describe the security functionality provided by each subsystem of the TSF.**

ADV_HLD.1.4C **The high-level design shall identify the interfaces of the subsystems of the TSF.**

ADV_HLD.1.5C **The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.**

Evaluator action elements:

ADV_HLD.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADV_HLD.1.2E **The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.**

ADV_HLD.2 Security enforcing high-level design

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_HLD.2.1D The developer shall provide the high-level design of the TSF.

Content and presentation of evidence elements:

ADV_HLD.2.1C The presentation of the high-level design shall be informal.

ADV_HLD.2.2C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.2.3C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.2.4C The high-level design shall identify the interfaces of the subsystems of the TSF.

ADV_HLD.2.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.2.6C **The high-level design shall describe the separation of the TSF into TSP enforcing and other subsystems.**

Evaluator action elements:

ADV_HLD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.2.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_HLD.3 Semiformal high-level design

Dependencies:

ADV_FSP.3 Semiformal security policy model

ADV_RCR.2 Semiformal correspondence demonstration

Developer action elements:

ADV_HLD.3.1D The developer shall provide the high-level design of the TSF.

Content and presentation of evidence elements:

ADV_HLD.3.1C The presentation of the high-level design shall be **semiformal**.

ADV_HLD.3.2C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.3.3C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.3.4C The high-level design shall identify the interfaces of the subsystems of the TSF.

ADV_HLD.3.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.3.6C The high-level design shall describe the separation of the TSF into TSP enforcing and other subsystems.

Evaluator action elements:

ADV_HLD.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.3.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_HLD.4 Semiformal high-level explanation

Dependencies:

ADV_FSP.3 Semiformal security policy model

ADV_RCR.2 Semiformal correspondence demonstration

Developer action elements:

ADV_HLD.4.1D The developer shall provide the high-level design of the TSF.

Content and presentation of evidence elements:

ADV_HLD.4.1C The presentation of the high-level design shall be semiformal.

ADV_HLD.4.2C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.4.3C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.4.4C The high-level design shall identify the interfaces of the subsystems of the TSF.

ADV_HLD.4.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.4.6C The high-level design shall describe the separation of the TSF into TSP enforcing and other subsystems.

ADV_HLD.4.7C **The evidence shall justify that the identified means of achieving separation, including any protection mechanisms, are sufficient to ensure a clear and effective separation of TSP enforcing from non-TSP enforcing functions.**

ADV_HLD.4.8C **The evidence shall justify that the TSF mechanisms are sufficient to implement the security functions.**

Evaluator action elements:

ADV_HLD.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.4.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_HLD.5 Formal high-level design

Dependencies:

ADV_FSP.4 Formal security policy model

ADV_RCR.3 Formal correspondence demonstration

Developer action elements:

ADV_HLD.5.1D The developer shall provide the high-level design of the TSF.

Content and presentation of evidence elements:

ADV_HLD.5.1C The presentation of the high-level design shall be **formal**.

ADV_HLD.5.2C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.5.3C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.5.4C The high-level design shall identify the interfaces of the subsystems of the TSF.

ADV_HLD.5.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.5.6C The high-level design shall describe the separation of the TSF into TSP enforcing and other subsystems.

ADV_HLD.5.7C The evidence shall justify that the identified means of achieving separation, including any protection mechanisms, are sufficient to ensure a clear and effective separation of TSP enforcing from non-TSP enforcing functions.

ADV_HLD.5.8C The evidence shall justify that the TSF mechanisms are sufficient to implement the security functions.

Evaluator action elements:

ADV_HLD.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.5.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_IMP Implementation representation

Objectives

259 The description of the implementation in the form of source code, firmware, hardware drawings, etc. captures the detailed internal workings of the TSF in support of analysis.

Component levelling

260 The components in this family are levelled on the basis of the completeness and structure of the implementation representations provided.

Application notes

261 The implementation representation is used to express the notion of the least abstract representation of the TSF, specifically the one that is used to create the TSF itself without further design refinement. Source code which is then compiled or a hardware drawing which is used to build the actual hardware are examples of parts of an implementation representation.

262 The evaluator of the TOE is expected to make determinations regarding the functional requirements in the ST relevant to the implementation. In the course of the implementation evaluation there are essentially three types of evaluator determination: specific functional requirements are met and no further work (e.g., with a more abstract representation of the TSF) is necessary; specific functional requirements are violated and the TOE fails to meet its requirements; and specific functional requirements have not been addressed and further analysis is necessary. However, since the implementation is the least abstract representation it is likely that further analysis cannot be performed, unless the TSF representations have not been evaluated in a usual order (i.e., most abstract to least abstract). If requirements are not addressed after the analysis of all TSF representations, this represents a failure of the TOE evaluation. Note that this more comprehensive failure determination requirement is realised in the Representation correspondence (ADV_RCR) family.

263 In all cases, it is important that the evaluator evaluates the TSF as a unit since in many cases the security functions must cooperate to meet specific functional requirements and also each security function must not interfere with the operation of any other security function.

264 It is expected that evaluators will use the implementation to directly support other evaluation activities (e.g., vulnerability analysis, test coverage analysis). It is expected that PP/ST authors will select a component that requires that the implementation is complete and comprehensible enough to address the needs of all other requirements included in the PP/ST.

ADV_IMP.1 Subset of the implementation of the TSF

Application notes

265 The PP/ST author should identify the subset of the implementation representation to be delivered. If a specific subset of the source code/hardware drawing to be delivered has not been specified by the PP/ST author, the evaluator has the option of requesting a subset of the source code/hardware drawings for analysis.

266 The intent is not an open ended invitation for the evaluator to demand implementation representations, but rather that the evaluator may request implementation representations that may support the demonstration that functional requirements have been met. For example, see the application notes for this family of assurance components.

Dependencies:

ADV_LLD.1 Descriptive low-level design

ADV_RCR.1 Informal correspondence demonstration

ALC_TAT.1 Well defined development tools

Developer action elements:

ADV_IMP.1.1D The developer shall provide the implementation representations for a selected subset of the TSF.

Content and presentation of evidence elements:

ADV_IMP.1.1C The implementation representations shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.

Evaluator action elements:

ADV_IMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_IMP.1.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_IMP.2 Implementation of the TSF

Dependencies:

ADV_LLD.1 Descriptive low-level design

ADV_RCR.1 Informal correspondence demonstration

ALC_TAT.2 Compliance with implementation standards

Developer action elements:

ADV_IMP.2.1D The developer shall provide the implementation representations for **the entire TSF**.

Content and presentation of evidence elements:

ADV_IMP.2.1C The implementation representations shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.2.2C **The implementation representations shall describe the relationships between all portions of the implementation.**

Evaluator action elements:

ADV_IMP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_IMP.2.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_IMP.3 Structured implementation of the TSF

Dependencies:

ADV_INT.1 Modularity

ADV_LLD.1 Descriptive low-level design

ADV_RCR.1 Informal correspondence demonstration

ALC_TAT.3 Compliance with implementation standards - all parts

Developer action elements:

ADV_IMP.3.1D The developer shall provide the implementation representations for the entire TSF.

Content and presentation of evidence elements:

ADV_IMP.3.1C The implementation representations shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.3.2C The implementation representations shall describe the relationships between all portions of the implementation.

ADV_IMP.3.3C **The implementation representations shall be structured into small and comprehensible sections.**

Evaluator action elements:

ADV_IMP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_IMP.3.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_INT TSF internals

Objectives

267 This family of components deals with the internal structure of the TSF. Requirements are established for modularity, the layering of the software architecture to separate levels of abstraction and minimisation of circular dependencies, and the minimisation from the TSF of software that is not TSP enforcing.

268 Modular design reduces the interdependence between elements of the TSF and thus reduces the risk that a change or error in one module will have effects throughout the TOE. Thus, a modular design provides the basis for determining the scope of interaction with other elements of the TSF, provides for increased assurance that unexpected effects do not occur, and also provides the basis for designing and evaluating test suites.

269 Design complexity affects how difficult it is to understand the design of the TOE. The simpler the design, the more assurance is gained that there are no hidden vulnerabilities in the design and that the high-level protection requirements are accurately and completely instantiated in the lower level design and the implementation.

270 Design complexity minimisation provides a part of the assurance that the code is understood; the less complex the code in the TSF, the greater the likelihood that the design of the TSF is comprehensible. Design complexity minimisation is a key characteristic of a reference validation mechanism.

Component levelling

271 The components in this family are levelled on the basis of the amount of structure and minimisation required.

Application notes

272 The term “relevant representation” is used in these components to cover the need for an evaluator to check for the appropriate issue (e.g., modularity, complexity) at whichever level of representation (e.g., high-level design, implementation) the requirements are being invoked.

273 The term “portions of the TSF” is used to represent parts of the TSF with a varying granularity based on the available TSF representations. The functional specification allows identification in terms of interfaces, the high-level design allows identification in terms of subsystems, the low-level design allows identification in terms of modules, and the implementation representation allows identification in terms of implementation units (e.g., source code files).

ADV_INT.1 Modularity

Dependencies:

ADV_IMP.1 Subset of the implementation of the TSF

ADV_LLD.1 Descriptive low-level design

Developer action elements:

ADV_INT.1.1D **The developer shall design the TSF in a modular fashion that avoids unnecessary interactions between the modules of the design.**

ADV_INT.1.2D **The developer shall provide an architectural description.**

Content and presentation of evidence elements:

ADV_INT.1.1C **The architectural description shall identify the modules of the TSF.**

ADV_INT.1.2C **The architectural description shall describe the purpose, interface, parameters, and effects of each module in the TSF.**

ADV_INT.1.3C **The architectural description shall describe how the TSF design provides for largely independent modules that avoid unnecessary interactions.**

Evaluator action elements:

ADV_INT.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADV_INT.1.2E **The evaluator shall check the relevant representations for compliance with the architectural description.**

ADV_INT.2 Layering

Application notes

274 This component introduces a reference monitor concept (i.e., small enough to be analysed) by requiring the minimisation of complexity of the portions of the TSF that enforce the access control and information flow policies identified in the TSP.

Dependencies:

ADV_IMP.1 Subset of the implementation of the TSF

ADV_LLD.1 Descriptive low-level design

Developer action elements:

ADV_INT.2.1D The developer shall design and structure the TSF in a modular **and layered** fashion that avoids unnecessary interactions between the modules of the design**, minimises mutual interactions between the layers of the design, and minimises the**

complexity of the portions of the TSF that enforce any access control and information flow policies.

ADV_INT.2.2D The developer shall provide an architectural description.

Content and presentation of evidence elements:

ADV_INT.2.1C The architectural description shall identify the modules of the TSF **and the portions of the TSF that enforce any access control and information flow policies**.

ADV_INT.2.2C The architectural description shall describe the purpose, interface, parameters, and effects of each module of the TSF.

ADV_INT.2.3C The architectural description shall describe how the TSF design provides for largely independent modules that avoid unnecessary interactions.

ADV_INT.2.4C **The architectural description shall describe the layering architecture.**

ADV_INT.2.5C **The architectural description shall show that mutual interactions have been eliminated or minimised, and justify those that remain.**

ADV_INT.2.6C **The architectural description shall describe how the portions of the TSF that enforce any access control and information flow policies have been structured to minimise complexity.**

Evaluator action elements:

ADV_INT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_INT.2.2E The evaluator shall check the relevant representations for compliance with the architectural description.

ADV_INT.3 Minimisation of Complexity

Application notes

275 This component requires that the reference monitor property “small enough to be analysed” is fully addressed. When this component is combined with the functional requirements FPT_RVM.1 and FPT_SEP.3, the reference monitor concept would be fully realised.

Dependencies:

ADV_IMP.2 Implementation of the TSF

ADV_LLD.1 Descriptive low-level design

Developer action elements:

ADV_INT.3.1D The developer shall design and structure the TSF in a modular and layered fashion that avoids unnecessary interactions between the modules of the design, minimises mutual interactions between the layers of the design, and minimises the complexity of the **entire TSF**.

ADV_INT.3.2D The developer shall provide an architectural description.

ADV_INT.3.3D **The developer shall design and structure the portions of the TSF that enforce any access control and information flow policies such that they are small enough to be analysed.**

ADV_INT.3.4D **The developer shall ensure that functions that are not relevant to TSP enforcement are excluded from the TSF.**

Content and presentation of evidence elements:

ADV_INT.3.1C The architectural description shall identify the modules of the TSF and the portions of the TSF that enforce any access control and information flow policies.

ADV_INT.3.2C The architectural description shall describe the purpose, interface, parameters, and side-effects of each module of the TSF.

ADV_INT.3.3C The architectural description shall describe how the TSF design provides for largely independent modules that avoid unnecessary interactions.

ADV_INT.3.4C The architectural description shall describe the layering architecture.

ADV_INT.3.5C The architectural description shall show that mutual interactions have been eliminated or minimised, and justify those that remain.

ADV_INT.3.6C The architectural description shall describe how the **entire TSF has** been structured to minimise complexity.

ADV_INT.3.7C **The architectural description shall justify the inclusion of any non TSP enforcing modules in the TSF.**

Evaluator action elements:

ADV_INT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_INT.3.2E The evaluator shall check the relevant representations for compliance with the architectural description.

ADV_INT.3.3E **The evaluator shall confirm that the portions of the TSF that enforce any access control and information flow policies are small enough to be analysed.**

ADV_LLD Low-level design

Objectives

276 The low-level design of a TOE provides a description of the internal workings of the TSF in terms of modules and their interrelationships and dependencies. The low-level design provides assurance that the TSF subsystems have been correctly and effectively refined.

277 For each module of the TSF, the low-level design describes its purpose, function, interfaces, dependencies, and the implementation of any TSP enforcing functions.

Component levelling

278 The components in this family are levelled on the basis of the degree of formalism required of the low-level design.

Application notes

279 In addition to the content indicated in the following requirements, the low-level design shall also include any additional specific detail specified by the documentation notes in the related functional components.

280 The evaluator of the TOE is expected to make determinations regarding the functional requirements in the ST relevant to the low-level design. In the course of the low-level design evaluation there are essentially three types of evaluator determination: specific functional requirements are met and no further work (e.g., with a less abstract representation of the TSF) is necessary; specific functional requirements are violated and the TOE fails to meet its requirements; and specific functional requirements have not been addressed and further analysis (of another TSF representation) is necessary. Whenever more analysis is necessary, the evaluator is expected to carry that information forward to the analysis of other TSF representations. If requirements are not addressed after the analysis of the last provided TSF representation, this also represents a failure of the TOE evaluation. Note that this more comprehensive failure determination requirement is realised in the Representation correspondence (ADV_RCR) family.

281 In all cases, it is important that the evaluator evaluates the TSF as a unit since in many cases the security functions must cooperate to meet specific functional requirements and also each security function must not interfere with the operation of any other security function.

282 The term “TSP enforcing function” refers to any function that contributes to TSP enforcement. The term “TSP enforcing modules” similarly refers to any module that contributes to TSP enforcement.

ADV_LLD.1 Descriptive low-level design

Application notes

283 Only representations for modules in the TSF need to be provided.

Dependencies:

ADV_HLD.1 Descriptive high-level design

ADV_RCR.1 Informal correspondence demonstration

Developer action elements:

ADV_LLD.1.1D The developer shall provide the low-level design of the TSF.

Content and presentation of evidence elements:

ADV_LLD.1.1C The presentation of the low-level design shall be informal.

ADV_LLD.1.2C The low-level design shall describe the TSF in terms of modules.

ADV_LLD.1.3C The low-level design shall describe the purpose of each module.

ADV_LLD.1.4C The low-level design shall define the interrelationships between the modules in terms of provided functionality and dependencies on other modules.

ADV_LLD.1.5C The low-level design shall describe the implementation of all TSP enforcing functions.

ADV_LLD.1.6C The low-level design shall describe the interfaces of each module in terms of their syntax and semantics.

ADV_LLD.1.7C The low-level design shall provide a demonstration that the TSF is completely represented.

ADV_LLD.1.8C The low-level design shall identify the interfaces of the modules of the TSF visible at the external interface of the TSF.

Evaluator action elements:

ADV_LLD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_LLD.1.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_LLD.2 Semiformal low-level design

Application notes

284 Only representations for modules in the TSF need to be provided.

Dependencies:

ADV_HLD.3 Semiformal high-level design

ADV_RCR.2 Semiformal correspondence demonstration

Developer action elements:

ADV_LLD.2.1D The developer shall provide the low-level design of the TSF.

Content and presentation of evidence elements:

ADV_LLD.2.1C The presentation of the low-level design shall be **semiformal**.

ADV_LLD.2.2C The low-level design shall describe the TSF in terms of modules.

ADV_LLD.2.3C The low-level design shall describe the purpose of each module.

ADV_LLD.2.4C The low-level design shall define the interrelationships between the modules in terms of provided functionality and dependencies on other modules.

ADV_LLD.2.5C The low-level design shall describe the implementation of all TSP enforcing functions.

ADV_LLD.2.6C The low-level design shall describe the interfaces of each module in terms of their syntax and semantics.

ADV_LLD.2.7C The low-level design shall provide a demonstration that the TSF is completely represented.

ADV_LLD.2.8C The low-level design shall identify the interfaces of the modules of the TSF visible at the external interface of the TSF.

ADV_LLD.2.9C **The low-level design shall describe the separation of the TSF into TSP enforcing and other modules.**

Evaluator action elements:

ADV_LLD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_LLD.2.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_LLD.3 Formal low-level design

Application notes

285 Only representations for modules in the TSF need to be provided.

Dependencies:

ADV_HLD.5 Formal high-level design

ADV_RCR.3 Formal correspondence demonstration

Developer action elements:

ADV_LLD.3.1D The developer shall provide the low-level design of the TSF.

Content and presentation of evidence elements:

ADV_LLD.3.1C The presentation of the low-level design shall be **formal**.

ADV_LLD.3.2C The low-level design shall describe the TSF in terms of modules.

ADV_LLD.3.3C The low-level design shall describe the purpose of each module.

ADV_LLD.3.4C The low-level design shall define the interrelationships between the modules in terms of provided functionality and dependencies on other modules.

ADV_LLD.3.5C The low-level design shall describe the implementation of all TSP enforcing functions.

ADV_LLD.3.6C The low-level design shall describe the interfaces of each module in terms of their syntax and semantics.

ADV_LLD.3.7C The low-level design shall provide a demonstration that the TSF is completely represented.

ADV_LLD.3.8C The low-level design shall identify the interfaces of the modules of the TSF visible at the external interface of the TSF.

ADV_LLD.3.9C The low-level design shall describe the separation of the TSF into TSP enforcing and other modules.

Evaluator action elements:

ADV_LLD.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_LLD.3.2E The evaluator shall determine if the functional requirements in the ST are addressed by the representation of the TSF.

ADV_RCR Representation correspondence

Objectives

286 The correspondence between the various representations (i.e. functional requirements expressed in the ST, functional specification, high-level design, low-level design, implementation) addresses the correct and complete instantiation of the requirements to the least abstract representation provided. This conclusion is achieved by step-wise refinement and the cumulative results of correspondence determinations between all adjacent abstractions of representation.

Component levelling

287 The components in this family are levelled on the basis of the level of rigour of the dependent representations, and thus reflect the level of rigour that can be obtained in the correspondence between the various abstractions of representation.

Application notes

288 The developer must demonstrate to the evaluator that the most detailed, or least abstract, representation of the TSF is an accurate, consistent, and complete instantiation of the functions expressed as functional requirements in the ST. This is accomplished by showing correspondence between adjacent representations at a commensurate level of rigour.

289 The evaluator must analyse each demonstration of correspondence between abstractions, as well as the results of the analysis of each TSF representation, and then make a determination as to whether the functional requirements in the ST have been satisfied.

290 This family of requirements is not intended to address correspondence relating to the TSP model or the TSP. Rather, as shown in Figure 5.4, it is intended to address correspondence between the requirements in the ST as well as the TOE summary specification, functional specification, high-level design, low-level design, and implementation representation.

ADV_RCR.1 Informal correspondence demonstration

Dependencies:

No dependencies.

Developer action elements:

ADV_RCR.1.1D **The developer shall provide evidence that the least abstract TSF representation provided is an accurate, consistent, and complete instantiation of the functional requirements expressed in the ST.**

Content and presentation of evidence elements:

ADV_RCR.1.1C **For each adjacent pair of TSF representations, the evidence shall demonstrate that all parts of the more abstract representation are refined in the less abstract representation.**

ADV_RCR.1.2C **For each adjacent pair of TSF representations, the demonstration of correspondence between the representations may be informal.**

Evaluator action elements:

ADV_RCR.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ADV_RCR.1.2E **The evaluator shall analyse the correspondence between the functional requirements expressed in the ST and the least abstract representation provided to ensure accuracy, consistency, and completeness.**

ADV_RCR.2 Semiformal correspondence demonstration

Dependencies:

No dependencies.

Developer action elements:

ADV_RCR.2.1D The developer shall provide evidence that the least abstract TSF representation provided is an accurate, consistent, and complete instantiation of the functional requirements expressed in the ST.

Content and presentation of evidence elements:

ADV_RCR.2.1C For each adjacent pair of TSF representations, the evidence shall demonstrate that all parts of the more abstract representation are refined in the less abstract representation.

ADV_RCR.2.2C For each adjacent pair of TSF representations, **where portions of both representations are at least semiformally specified,** the demonstration of correspondence between **those portions of** the representations **shall** be **semiformal**.

ADV_RCR.2.3C **For each adjacent pair of TSF representations, where portions of either representation are informally specified the demonstration of correspondence between those portions of the representations may be informal.**

Evaluator action elements:

ADV_RCR.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_RCR.2.2E The evaluator shall analyse the correspondence between the functional requirements expressed in the ST and the least abstract representation provided to ensure accuracy, consistency, and completeness.

ADV_RCR.3 Formal correspondence demonstration

Application notes

291 The developer must either demonstrate or prove correspondence, as described in the requirements below, commensurate with the level of rigour of presentation style. For example, correspondence must be proven when corresponding representations are formally specified.

Dependencies:

No dependencies.

Developer action elements:

ADV_RCR.3.1D The developer shall provide evidence that the least abstract TSF representation provided is an accurate, consistent, and complete instantiation of the functional requirements expressed in the ST.

ADV_RCR.3.2D **For those corresponding portions of representations that are formally specified, the developer shall prove that correspondence.**

Content and presentation of evidence elements:

ADV_RCR.3.1C For each adjacent pair of TSF representations, the evidence shall **prove or** demonstrate that all parts of the more abstract representation are refined in the less abstract representation.

ADV_RCR.3.2C For each adjacent pair of TSF representations, where portions of **one** representation are **semiformally specified and the other** at least semi-formally specified, the demonstration of correspondence between those portions of the representations shall be semiformal.

ADV_RCR.3.3C For each adjacent pair of TSF representations, where portions of either representation are informally specified the demonstration of correspondence between those portions of the representations may be informal.

ADV_RCR.3.4C **For each adjacent pair of TSF representations, where portions of both representations are formally specified the proof of correspondence between those portions of the representations shall be formal.**

Evaluator action elements:

ADV_RCR.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_RCR.3.2E The evaluator shall analyse the correspondence between the functional requirements expressed in the ST and the least abstract representation provided to ensure accuracy, consistency, and completeness.

ADV_RCR.3.3E **The evaluator shall determine the accuracy of the proofs of correspondence by selectively verifying the formal analysis.**

Class AGD

Guidance documents

292 The guidance documents class provides the requirements for user and administrator guidance documentation. For the secure installation and use of the TOE it is necessary to describe all relevant aspects for the secure application of the TOE.

293 Figure 5.5 shows the families within this class, and the hierarchy of components within the families.

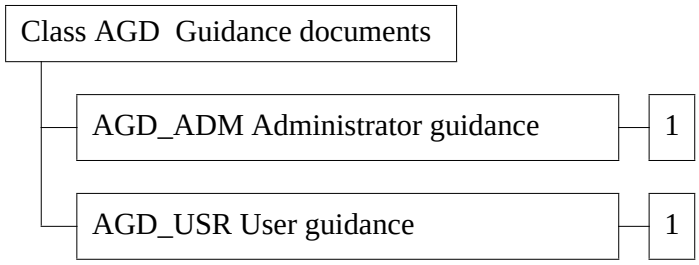

Figure 5.5 -Guidance documents class decomposition

AGD_ADM Administrator guidance

Objectives

294 Administrator guidance refers to written material that is intended to be used by those persons responsible for configuring, maintaining, and administering the TOE in a correct manner for maximum security. Because the secure operation of the TOE is dependent upon the correct performance of the TSF, persons responsible for performing these functions are trusted by the TSF. Administrator guidance is intended to help administrators understand the security functions provided by the TOE, including both those functions that require the administrator to perform security-critical actions and those functions that provide security-critical information.

Component levelling

295 This family contains only one component.

Application notes

296 The requirements AGD_ADM.1.2C and AGD_ADM.1.11C encompass the aspect that any warnings to the users of a TOE with regard to the TOE security environment and the security objectives described in the PP/ST are appropriately covered in the administrator guidance.

297 The PP/ST author should review the functional components of the PP/ST for guidance on administrator documentation. Those application notes that are relevant to administrator guidance for understanding and proper application of the security functions should be considered for inclusion in the administrator guidance requirements. An example of an administrator guidance document is a reference manual.

AGD_ADM.1 Administrator guidance

Dependencies:

ADV_FSP.1 TOE and security policy

Developer action elements:

AGD_ADM.1.1D **The developer shall provide administrator guidance addressed to system administrative personnel.**

Content and presentation of evidence elements:

AGD_ADM.1.1C **The administrator guidance shall describe how to administer the TOE in a secure manner.**

AGD_ADM.1.2C **The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.**

AGD_ADM.1.3C **The administrator guidance shall contain guidelines on the consistent and effective use of the security functions within the TSF.**

AGD_ADM.1.4C **The administrator guidance shall describe the difference between two types of functions: those which allow an administrator to control security parameters, and those which allow the administrator to obtain information only.**

AGD_ADM.1.5C **The administrator guidance shall describe all security parameters under the administrator's control.**

AGD_ADM.1.6C **The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.**

AGD_ADM.1.7C **The administrator guidance shall contain guidelines on how the security functions interact.**

AGD_ADM.1.8C **The administrator guidance shall contain instructions regarding how to configure the TOE.**

AGD_ADM.1.9C **The administrator guidance shall describe all configuration options that may be used during secure installation of the TOE.**

AGD_ADM.1.10C **The administrator guidance shall describe details, sufficient for use, of procedures relevant to the administration of security.**

AGD_ADM.1.11C **The administrator guidance shall be consistent with all other documents supplied for evaluation.**

Evaluator action elements:

AGD_ADM.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

AGD_ADM.1.2E **The evaluator shall confirm that the installation procedures result in a secure configuration.**

AGD_USR User guidance

Objectives

298 User guidance refers to written material that is intended to be used by nonadministrative (human) users of the TOE. User guidance describes the security functions provided by the TSF and provides instructions and guidelines, including warnings, for its secure use.

299 The user guidance provides a basis for assumptions about the use of the TOE and a measure of confidence that non-malicious users and application providers will understand the secure operation of the TOE and will use it as intended.

Component levelling

300 This family contains only one component.

Application notes

301 The requirement AGD_USR.1.3.C and AGD_USR.1.5C encompass the aspect that any warnings to the users of a TOE with regard to the TOE security environment and the security objectives described in the PP/ST are appropriately covered in the user guidance.

302 The PP/ST author should review the functional components of the PP/ST for guidance on user documentation. Those application notes that are relevant to user guidance aimed at the understanding and proper use of the security functions should be considered for inclusion in the user guidance requirements. Examples of user guidance are reference manuals, user guides, and on-line help.

AGD_USR.1 User guidance

Dependencies:

ADV_FSP.1 TOE and security policy

Developer action elements:

AGD_USR.1.1D **The developer shall provide user guidance.**

Content and presentation of evidence elements:

AGD_USR.1.1C **The user guidance shall describe the TSF and interfaces available to the user.**

AGD_USR.1.2C **The user guidance shall contain guidelines on the use of security functions provided by the TOE.**

AGD_USR.1.3C **The user guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.**

AGD_USR.1.4C **The user guidance shall describe the interaction between user-visible security functions.**

AGD_USR.1.5C **The user guidance shall be consistent with all other documentation delivered for evaluation.**

Evaluator action elements:

AGD_USR.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

Class ALC

Life cycle support

303 Life-cycle support is an aspect of establishing discipline and control in the processes of refinement of the TOE during development and maintenance. Confidence in the correspondence between the TOE security requirements and the TOE is greater if security analysis and the production of the evidence are done on a regular basis as an integral part of the development and maintenance activities.

304 Figure 5.6 shows the families within this class, and the hierarchy of components within the families.

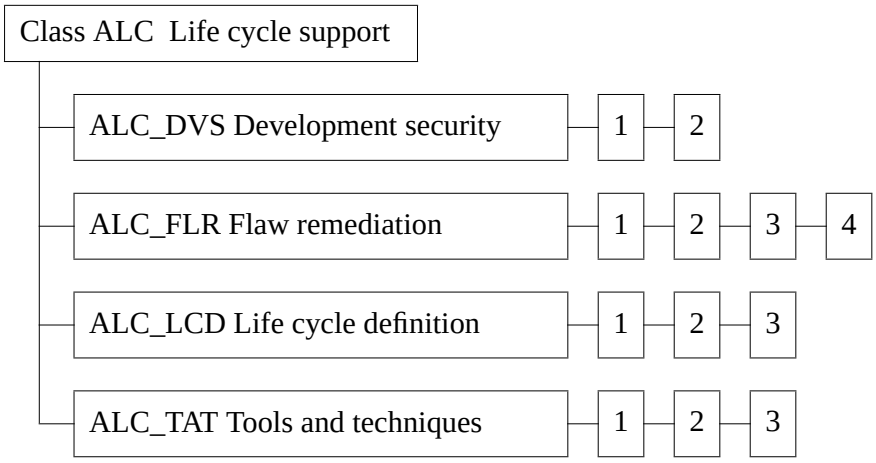

Figure 5.6 -Life-cycle support class decomposition

ALC_DVS Development security

Objectives

305 Development security is concerned with physical, procedural, personnel, and other security measures that may be used in the development environment to protect the TOE. It includes the physical security of the development location and any procedures used to select development staff.

Component levelling

306 The components in this family are levelled on the basis of whether justification of the sufficiency of the security measures is required.

Application notes

307 The evaluator should decide whether there is a need for visiting the user's site in order to confirm that the requirements of this family are met.

ALC_DVS.1 Identification of security measures

Dependencies:

No dependencies.

Developer action elements:

ALC_DVS.1.1D **The developer shall produce development security documentation.**

Content and presentation of evidence elements:

ALC_DVS.1.1C **The development security documentation shall describe the physical, procedural, personnel, and other security measures that are used to protect the confidentiality and integrity of the TOE during its development.**

ALC_DVS.1.2C **The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.**

Evaluator action elements:

ALC_DVS.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ALC_DVS.1.2E **The evaluator shall check whether the security measures are being applied.**

ALC_DVS.2 Sufficiency of security measures

Dependencies:

No dependencies.

Developer action elements:

ALC_DVS.2.1D The developer shall produce development security documentation.

Content and presentation of evidence elements:

ALC_DVS.2.1C The development security documentation shall describe the physical, procedural, personnel, and other security measures that are used to protect the confidentiality and integrity of the TOE during its development.

ALC_DVS.2.2C The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.

ALC_DVS.2.3C **The evidence shall justify that the security measures are sufficient to protect the confidentiality and integrity of the TOE.**

Evaluator action elements:

ALC_DVS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_DVS.2.2E The evaluator shall check whether the security measures are being applied.

ALC_FLR Flaw remediation

Objectives

308 Flaw remediation requires that discovered flaws be tracked and corrected by the developer. Although future compliance with flaw remediation procedures cannot be determined at the time of the TOE evaluation, it is possible to evaluate the policies and procedures that a developer has in place to track and correct flaws, and to distribute the flaw information and corrections.

Component levelling

309 The components in this family are levelled on the basis of the increasing extent in scope of the flaw remediation procedures and the rigour of the flaw remediation policies.

Application notes

310 The PP/ST author should consider whether it would be useful to introduce the assurance provided by a flaw remediation component into the PP/ST. This should receive special attention as no flaw remediation component is included in any EAL and the absence of such components decreases the assurance that the TOE received is well maintained and supported in the future. Specifically, security flaws may not be properly corrected and corrections may not be distributed. In considering which flaw remediation component to select, the selected EAL and intended application of the TOE should be primary factors. In general, higher FLR components are more appropriate for the higher EALs and for very sensitive applications.

ALC_FLR.1 Basic flaw remediation

Dependencies:

No dependencies.

Developer action elements:

ALC_FLR.1.1D **The developer shall document the flaw remediation procedures.**

Content and presentation of evidence elements:

ALC_FLR.1.1C **The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.**

ALC_FLR.1.2C **The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.**

ALC_FLR.1.3C **The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.**

ALC_FLR.1.4C **The flaw remediation procedures documentation shall describe the methods used to provide flaw information and corrections to TOE users.**

Evaluator action elements:

ALC_FLR.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ALC_FLR.2 Flaw reporting procedures

Dependencies:

AGD_ADM.1 Administrator guidance

Developer action elements:

ALC_FLR.2.1D The developer shall document the flaw remediation procedures.

ALC_FLR.2.2D **The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.**

Content and presentation of evidence elements:

ALC_FLR.2.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.2.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.2.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.2.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information and corrections to TOE users.

ALC_FLR.2.5C **The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.**

Evaluator action elements:

ALC_FLR.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_FLR.3 Systematic flaw remediation

Dependencies:

AGD_ADM.1 Administrator guidance

Developer action elements:

ALC_FLR.3.1D The developer shall document the flaw remediation procedures.

ALC_FLR.3.2D The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.

ALC_FLR.3.3D **The developer shall designate one or more specific points of contact for user reports and inquiries about security issues involving the TOE.**

Content and presentation of evidence elements:

ALC_FLR.3.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.3.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.3.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.3.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information and corrections to TOE users.

ALC_FLR.3.5C The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.

ALC_FLR.3.6C **The flaw remediation procedures shall include a procedure for the automatic distribution of security flaw reports and the associated corrections to registered users who might be affected by the security flaw.**

Evaluator action elements:

ALC_FLR.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_FLR.4 Timely flaw remediation

Dependencies:

AGD_ADM.1 Administrator guidance

Developer action elements:

ALC_FLR.4.1D The developer shall document the flaw remediation procedures.

ALC_FLR.4.2D The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.

ALC_FLR.4.3D The developer shall designate one or more specific points of contact for user reports and inquiries about security issues involving the TOE.

Content and presentation of evidence elements:

ALC_FLR.4.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.4.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.4.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.4.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information and corrections to TOE users.

ALC_FLR.4.5C The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.

ALC_FLR.4.6C The flaw remediation procedures shall include a procedure **requiring timely responses** for the automatic distribution of security flaw reports and the associated corrections to registered users who might be affected by the security flaw.

Evaluator action elements:

ALC_FLR.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_LCD Life cycle definition

Objectives

311 Poorly controlled development and maintenance can result in a flawed implementation of a TOE (or a TOE that does not meet all of its security requirements). This, in turn, results in security violations. Therefore, it is important that a model for the development and maintenance of a TOE be established as early as possible in the TOE's life-cycle.

312 Using a model for the development and maintenance of a TOE does not guarantee that the TOE will be free of flaws, nor does it guarantee that the TOE will meet all of its security functional requirements. It is possible that the model chosen was insufficient or inadequate and therefore no benefits in the quality of the TOE could be observed. Using a life-cycle model that has been approved by some group of experts (e.g., academic experts, standards bodies) improves the chances that the development and maintenance models will contribute to the overall quality of the TOE.

Component levelling

313 The components in this family are levelled on the basis of increasing requirements for standardisation and measurability of the life-cycle model, and for compliance with that model.

Application notes

314 Although life-cycle definition deals with the maintenance of the TOE and hence with aspects becoming relevant after the completion of the evaluation, its evaluation adds assurance through an analysis the life-cycle information for the TOE provided at the time of the evaluation.

315 A life-cycle model encompasses the procedures, tools and techniques used to develop and maintain the TOE.

316 A standardised life-cycle model is a model that has been approved by some group of experts (e.g., academic experts, standards bodies).

317 A measurable life-cycle model is a model with some arithmetic parameters so that e.g. the coding standards can be measured.

ALC_LCD.1 Developer defined life-cycle model

Dependencies:

No dependencies.

Developer action elements:

ALC_LCD.1.1D **The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.**

ALC_LCD.1.2D **The developer shall produce life-cycle definition documentation.**

Content and presentation of evidence elements:

ALC_LCD.1.1C **The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.**

Evaluator action elements:

ALC_LCD.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ALC_LCD.2 Standardised life-cycle model

Dependencies:

No dependencies.

Developer action elements:

ALC_LCD.2.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.

ALC_LCD.2.2D The developer shall produce life-cycle definition documentation.

ALC_LCD.2.3D **The developer shall use a standardised life-cycle model to develop and maintain the TOE.**

Content and presentation of evidence elements:

ALC_LCD.2.1C The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.

ALC_LCD.2.2C **The life-cycle definition documentation shall explain why the model was chosen and how it is used to develop and maintain the TOE.**

ALC_LCD.2.3C **The life-cycle definition documentation shall demonstrate compliance with the standardised life-cycle model.**

Evaluator action elements:

ALC_LCD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_LCD.3 Measurable life-cycle model

Dependencies:

No dependencies.

Developer action elements:

ALC_LCD.3.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.

ALC_LCD.3.2D The developer shall produce life-cycle definition documentation.

ALC_LCD.3.3D The developer shall use a standardised **and measurable** life-cycle model to develop and maintain the TOE.

Content and presentation of evidence elements:

ALC_LCD.3.1C The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.

ALC_LCD.3.2C The life-cycle definition documentation shall explain why the model was chosen and how it is used to develop and maintain the TOE.

ALC_LCD.3.3C The life-cycle definition documentation shall demonstrate compliance with the standardised **and measurable** life-cycle model.

Evaluator action elements:

ALC_LCD.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_TAT Tools and techniques

Objectives

318 Tools and techniques is an aspect of selecting tools which are used to develop, analyse and implement the TOE. It includes requirements to prevent ill-defined, inconsistent or incorrect development tools from being used to develop the TOE. This includes, but is not limited to programming languages, documentation, implementation standards, and other parts of the TOE like supporting runtime libraries.

Component levelling

319 The components in this family are levelled on the basis of increasing requirements on the description and scope of the implementation standards and the documentation of implementation dependent options.

Application notes

320 There is a requirement for well-defined development tools. These are tools which have been shown to be well understood and applicable without the need for intensive further clarification. For example, programming languages and computer aided design (CAD) systems that are based on an a standard published by standards bodies are considered to be well-defined.

321 Tools and techniques distinguishes between the implementation standards applied by the developer and the implementation standards for “all parts of the TOE” which additionally includes third party software, hardware, or firmware.

322 The requirement in ALC_TAT.1.2C is specifically applicable to programming languages so as to ensure that all statements in the source code have an unambiguous meaning.

ALC_TAT.1 Well defined development tools

Dependencies:

No dependencies.

Developer action elements:

ALC_TAT.1.1D **The developer shall identify the development tools being used for the TOE.**

ALC_TAT.1.2D **The developer shall document the selected implementation dependent options of the development tools.**

Content and presentation of evidence elements:

ALC_TAT.1.1C **Any development tools used for implementation shall be well-defined.**

ALC_TAT.1.2C **The documentation of the development tools shall unambiguously define the meaning of all statements used in the implementation.**

Evaluator action elements:

ALC_TAT.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ALC_TAT.2 Compliance with implementation standards

Dependencies:

ADV_IMP.1 Subset of the implementation of the TSF

Developer action elements:

ALC_TAT.2.1D The developer shall identify the development tools being used for the TOE.

ALC_TAT.2.2D The developer shall document the selected implementation dependent options of the development tools.

ALC_TAT.2.3D **The developer shall describe the implementation standards to be applied.**

Content and presentation of evidence elements:

ALC_TAT.2.1C Any development tools used for implementation shall be well-defined.

ALC_TAT.2.2C The documentation of the development tools shall unambiguously define the meaning of all statements used in the implementation.

Evaluator action elements:

ALC_TAT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_TAT.2.2E **The evaluator shall confirm that the implementation standards have been applied.**

ALC_TAT.3 Compliance with implementation standards - all parts

Dependencies:

ADV_IMP.1 Subset of the implementation of the TSF

Developer action elements:

ALC_TAT.3.1D The developer shall identify the development tools being used for the TOE.

ALC_TAT.3.2D The developer shall document the selected implementation dependent options of the development tools.

ALC_TAT.3.3D The developer shall describe the implementation standards **for all parts of the TOE**.

Content and presentation of evidence elements:

ALC_TAT.3.1C Any development tools used for implementation shall be well-defined.

ALC_TAT.3.2C The documentation of the development tools shall unambiguously define the meaning of all statements used in the implementation.

Evaluator action elements:

ALC_TAT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_TAT.3.2E The evaluator shall confirm that the implementation standards have been applied.

Class ATE

Tests

323 The class “Tests” encompasses four families: coverage (ATE_COV), depth (ATE_DPT), independent testing (e.g., functional testing performed by evaluators) (ATE_IND), and functional tests (ATE_FUN). Testing establishes that the TSF exhibits the properties necessary to satisfy the functional requirements of the PP/ ST. Testing provides assurance that the TSF satisfies at least the security functional requirements, although it cannot establish that the TSF does no more than what was specified. Testing may also be directed toward the internals of the TSF, such as the testing of subsystems and modules against their specifications.

324 The aspects of coverage and depth have been separated from functional tests for reasons of increased flexibility in applying the components of the families. However, the requirements in these three families are intended to be applied together.

325 The independent testing has dependencies on the other families to provide the necessary information to support the requirements, but is primarily concerned with independent evaluator actions.

326 This class does not address penetration testing, which is directed toward finding vulnerabilities that enable a user to violate the security policy. Penetration testing is addressed separately as an aspect of vulnerability assessment in the class AVA.

327 Figure 5.7 shows the families within this class, and the hierarchy of components within the families.

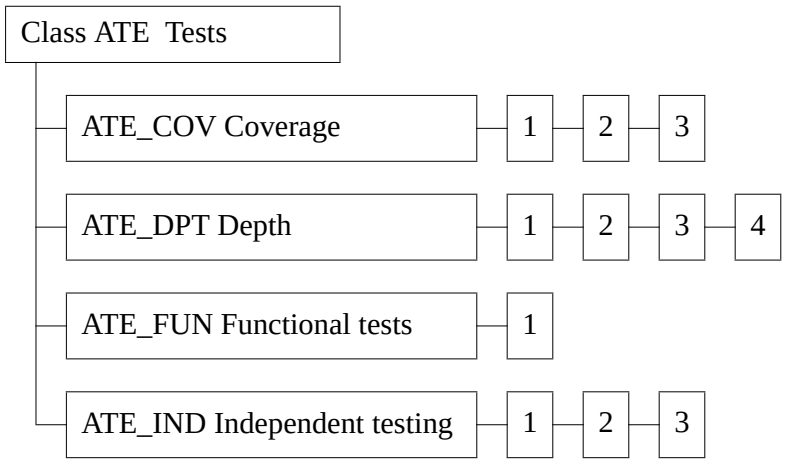

Figure 5.7 -Tests class decomposition

ATE_COV Coverage

Objectives

328 This family addresses those aspects of testing that deal with completeness of testing. That is, it addresses the extent to which the TOE security functions are tested, whether or not the testing is sufficiently extensive to demonstrate that the TSF operates as specified, and whether or not the order in which testing proceeds correctly accounts for functional dependencies between the portions of the TOE being tested.

Component levelling

329 The components in this family are levelled on the basis of increasing rigour of the analysis of the sufficiency of the tests to demonstrate that the TOE operates as stated.

Application notes

330 The specific documentation required by the coverage components will be determined, in most cases, by the documentation stipulated in the level of ATE_FUN that is specified. However, the PP/ST author will need to give consideration to the proper set of test evidence and documentation required.

ATE_COV.1 Complete coverage - informal

Objectives

331 In this component, the objective is that testing completely address the security functions.

Application notes

332 While the testing objective is to completely cover the TSF, there is no more than informal explanation to support this assertion.

Dependencies:

ADV_FSP.1 TOE and security policy

ATE_FUN.1 Functional testing

Developer action elements:

ATE_COV.1.1D **The developer shall provide an analysis of the test coverage.**

Content and presentation of evidence elements:

ATE_COV.1.1C **The analysis of the test coverage shall demonstrate that the tests identified in the test documentation cover the TSF.**

Evaluator action elements:

ATE_COV.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ATE_COV.2 Complete coverage - rigorous

Objectives

333 The objective is that testing completely address the security functions.

334 In this component, the objective is to ensure that there is a detailed correspondence between the tests and the security functions.

Application notes

335 The analysis of the test coverage in support of the detailed correspondence can be informal.

Dependencies:

ADV_FSP.1 TOE and security policy

ATE_FUN.1 Functional testing

Developer action elements:

ATE_COV.2.1D The developer shall provide an analysis of the test coverage.

Content and presentation of evidence elements:

ATE_COV.2.1C The analysis of the test coverage shall demonstrate that the tests identified in the test documentation cover the TSF.

ATE_COV.2.2C **The analysis of the test coverage shall demonstrate the correspondence between the security functions and the tests identified in the test documentation.**

Evaluator action elements:

ATE_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_COV.3 Ordered testing

Objectives

336 The objective is that testing completely address the security functions.

337 The objective is to ensure that there is a detailed correspondence between the tests and the security functions.

338 In this component, an additional objective is detailed justification that testing is structured such as to avoid circular arguments about the correctness of the portions of the TOE being tested.

Application notes

339 Ordering dependencies between tests can be of different forms e.g., test A provides a result to test B; test A cannot run before test B, since it breaks something required by test B; test failure in test B might be because of a failure in "untested" test A.

Dependencies:

ADV_FSP.1 TOE and security policy

ATE_FUN.1 Functional testing

Developer action elements:

ATE_COV.3.1D The developer shall provide an analysis of the test coverage.

ATE_COV.3.2D **The developer shall provide an analysis of ordering dependencies of tests.**

Content and presentation of evidence elements:

ATE_COV.3.1C The analysis of the test coverage shall demonstrate that the tests identified in the test documentation cover the TSF.

ATE_COV.3.2C The analysis of the test coverage shall demonstrate the correspondence between the security functions and the tests identified in the test documentation.

ATE_COV.3.3C **The analysis documentation shall justify that the correspondence is complete.**

ATE_COV.3.4C **The analysis documentation shall describe the ordering dependencies of tests.**

ATE_COV.3.5C **The analysis documentation shall justify that the test plans and procedures are consistent with the ordering dependencies of tests.**

Evaluator action elements:

ATE_COV.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_DPT Depth

Objectives

340 The components in this family deal with the level of detail to which the TOE is tested. Testing of security functions is based upon increasing depth of information derived from analysis of the representations.

341 The objective is to counter the risk of missing an error in the development of the TOE. Additionally, the components of this family, especially as testing is more concerned with the internals of the TOE, are more likely to discover any malicious code that has been inserted.

Component levelling

342 The components in this family are levelled on the increasing level of detail provided in the TSF representations, from the most abstract (functional specification) to the least abstract (implementation). This levelling reflects the representations presented in the ADV class.

Application notes

343 The specific amount and type of documentation and evidence will, in general, be determined by that required by level of ATE_FUN selected. However, the PP/ST author will need to give consideration to the proper set of test evidence and documentation required.

ATE_DPT.1 Testing - functional specification

Objectives

344 The functional specification of a TOE provides a high level description of the external workings of the TSF. Testing at the level of the functional specification, in order to demonstrate the presence of any flaws, provides assurance that the TSF functional specification has been correctly realised.

Application notes

345 The functional specification representation is used to express the notion of the most abstract representation of the TSF.

Dependencies:

ADV_FSP.1 TOE and security policy

ATE_FUN.1 Functional testing

Developer action elements:

ATE_DPT.1.1D **The developer shall provide the analysis of the depth of testing.**

Content and presentation of evidence elements:

ATE_DPT.1.1C **The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TOE operates in accordance with the functional specification of the TSF.**

Evaluator action elements:

ATE_DPT.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ATE_DPT.2 Testing - high level design

Objectives

346 The functional specification of a TOE provides a high level description of the external workings of the TSF. Testing at the level of the functional specification, in order to demonstrate the presence of any flaws, provides assurance that the TSF functional specification has been correctly realised.

347 The subsystems of a TOE provide a high level description of the internal workings of the TSF. Testing at the level of the subsystems, in order to demonstrate the presence of any flaws, provides assurance that the TSF subsystems have been correctly realised.

Application notes

348 The functional specification representation is used to express the notion of the most abstract representation of the TSF.

349 The developer is expected to describe the testing of the high level design of the TSF in terms of “subsystems”. The term “subsystem” is used to express the notion of decomposing the TSF into a relatively small number of parts. While the developer is not required to actually have “subsystems”, the developer is expected to represent a similar notion of decomposition.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ATE_FUN.1 Functional testing

Developer action elements:

ATE_DPT.2.1D The developer shall provide the analysis of the depth of testing.

Content and presentation of evidence elements:

ATE_DPT.2.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TOE operates in accordance with the functional specification**, and high level design** of the TSF.

Evaluator action elements:

ATE_DPT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_DPT.3 Testing - low level design

Objectives

350 The functional specification of a TOE provides a high level description of the external workings of the TSF. Testing at the level of the functional specification, in order to demonstrate the presence of any flaws, provides assurance that the TSF functional specification has been correctly realised.

351 The subsystems of a TOE provide a high level description of the internal workings of the TSF. Testing at the level of the subsystems, in order to demonstrate the presence of any flaws, provides assurance that the TSF subsystems have been correctly realised.

352 The modules of a TOE provide a description of the internal workings of the TSF. Testing at the level of the modules, in order to demonstrate the presence of any flaws, provides assurance that the TSF modules have been correctly realised.

Application notes

353 The functional specification representation is used to express the notion of the most abstract representation of the TSF.

354 The developer is expected to describe the testing of the high level design of the TSF in terms of “subsystems”. The term “subsystem” is used to express the notion of decomposing the TSF into a relatively small number of parts. While the developer is not required to actually have “subsystems”, the developer is expected to represent a similar notion of decomposition.

355 The developer is expected to describe the testing of the low level design of the TSF in terms of “modules”. The term “modules” is used to express the notion of decomposing each of the “subsystems” of the TSF into a relatively small number of parts. While the developer is not required to actually have “modules”, the developer is expected to represent a similar notion of decomposition.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ADV_LLD.1 Descriptive low-level design

ATE_FUN.1 Functional testing

Developer action elements:

ATE_DPT.3.1D The developer shall provide the analysis of the depth of testing.

Content and presentation of evidence elements:

ATE_DPT.3.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TOE operates in accordance with the functional specification, high level design**, and low level design** of the TSF.

Evaluator action elements:

ATE_DPT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_DPT.4 Testing - implementation

Objectives

356 The functional specification of a TOE provides a high level description of the external workings of the TSF. Testing at the level of the functional specification, in order to demonstrate the presence of any flaws, provides assurance that the TSF functional specification has been correctly realised.

357 The subsystems of a TOE provide a high level description of the internal workings of the TSF. Testing at the level of the subsystems, in order to demonstrate the presence of any flaws, provides assurance that the TSF subsystems have been correctly realised.

358 The modules of a TOE provide a description of the internal workings of the TSF. Testing at the level of the modules, in order to demonstrate the presence of any flaws, provides assurance that the TSF modules have been correctly realised.

359 The implementation representation of a TOE provides a detailed description of the internal workings of the TSF. Testing at the level of the implementation, in order to demonstrate the presence of any flaws, provides assurance that the TSF implementation has been correctly realised.

Application notes

360 The functional specification representation is used to express the notion of the most abstract representation of the TSF.

361 The developer is expected to describe the testing of the high level design of the TSF in terms of “subsystems”. The term “subsystem” is used to express the notion of

decomposing the TSF into a relatively small number of parts. While the developer is not required to actually have “subsystems”, the developer is expected to represent a similar notion of decomposition.

362 The developer is expected to describe the testing of the low level design of the TSF in terms of “modules”. The term “modules” is used to express the notion of decomposing each of the “subsystems” of the TSF into a relatively small number of parts. While the developer is not required to actually have “modules”, the developer is expected to represent a similar notion of decomposition.

363 The implementation representation is used to express the notion of the least abstract representation of the TSF, specifically the one which is used to generate the TSF itself (e.g., source code which is then compiled).

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ADV_IMP.2 Implementation of the TSF

ADV_LLD.1 Descriptive low-level design

ATE_FUN.1 Functional testing

Developer action elements:

ATE_DPT.4.1D The developer shall provide the analysis of the depth of testing.

Content and presentation of evidence elements:

ATE_DPT.4.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TOE operates in accordance with the functional specification, high level design, low level design**, and implementation** of the TSF.

Evaluator action elements:

ATE_DPT.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_FUN Functional tests

Objectives

364 Functional testing establishes that the TSF exhibits the properties necessary to satisfy the functional requirements of its PP/ST. Functional testing provides assurance that the TSF satisfies at least the security functional requirements, although it cannot establish that the TSF does no more than what was specified. The family “Functional tests” is focused on the type and amount of documentation or support tools required, and what is to be demonstrated through testing.

365 This family contributes to providing assurance that the likelihood of undiscovered flaws is relatively small.

Component levelling

366 This family contains only one component.

Application notes

367 Procedures for performing tests are expected to provide instructions for using test programs and test suites, including the test environment, test conditions, test data parameters and values. The test procedures should also show how the test results is derived from the test inputs.

368 The developer shall eliminate all security relevant flaws discovered during testing.

369 The developer shall test the TSF to determine that no new security relevant flaws have been introduced as a result of eliminating discovered security relevant flaws.

ATE_FUN.1 Functional testing

Objectives

370 The objective is for the developer to demonstrate that all security functions perform as specified. The developer is required to perform testing and to provide test documentation.

Dependencies:

ATE_COV.1 Complete coverage - informal

ATE_DPT.1 Testing - functional specification

Developer action elements:

ATE_FUN.1.1D **The developer shall test the TSF and document the results.**

ATE_FUN.1.2D **The developer shall provide test documentation.**

Content and presentation of evidence elements:

ATE_FUN.1.1C **The test documentation shall consist of test plans, test procedure descriptions, and test results.**

ATE_FUN.1.2C **The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.**

ATE_FUN.1.3C **The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function.**

ATE_FUN.1.4C **The test results in the test documentation shall show the expected results of each test.**

ATE_FUN.1.5C **The test results from the developer execution of the tests shall demonstrate that each security function operates as specified.**

Evaluator action elements:

ATE_FUN.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ATE_IND Independent testing

Objectives

371 The objective is to demonstrate that the security functions perform as specified.

372 Additionally, an objective is to counter the risk of an incorrect assessment of the test outcomes on the part of the developer which results in the incorrect implementation of the specifications, or overlooks code that is non-compliant with the specifications.

Component levelling

373 Levelling is based upon the amount of test documentation, test support and the amount of evaluator testing.

Application notes

374 The testing specified in this family can be performed by a party other than the evaluator (e.g., an independent laboratory, an objective consumer organisation).

375 This family deals with the degree to which there is independent functional testing of the TOE. Independent functional testing may take the form of repeating the developer's functional tests, in whole or in part. It may also take the form of the augmentation of the developer's functional tests, either to extend the scope or the depth of the developer's tests.

ATE_IND.1 Independent testing - conformance

Objectives

376 In this component, the objective is to demonstrate that the security functions perform as specified.

Application notes

377 The suitability of the TOE for testing is based on the access to the TOE, and the supporting documentation and information required to run tests. The need for documentation is supported by the dependencies to other assurance families.

378 Additionally, suitability of the TOE for testing may be based on other considerations e.g., the version of the TOE submitted by the developer is not the final version.

Dependencies:

ADV_FSP.1 TOE and security policy

AGD_USR.1 User guidance

AGD_ADM.1 Administrator guidance

Developer action elements:

ATE_IND.1.1D **The developer shall provide the TOE for testing.**

Content and presentation of evidence elements:

ATE_IND.1.1C **The TOE shall be suitable for testing.**

Evaluator action elements:

ATE_IND.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

ATE_IND.1.2E **The evaluator shall test the TSF to confirm that the TSF operates as specified.**

ATE_IND.2 Independent testing - sample

Objectives

379 The objective is to demonstrate that the security functions perform as specified.

380 In this component, the objective is to select and repeat a sample of the developer testing.

Application notes

381 The suitability of the TOE for testing is based on the access to the TOE, and the supporting documentation and information required to run tests. The need for documentation is supported by the dependencies to other assurance families.

382 Additionally, suitability of the TOE for testing may be based on other considerations e.g., the version of the TOE submitted by the developer is not the final version.

383 The developer is required to perform testing and to provide test documentation and test results. This is addressed by the ATE_FUN family.

384 Testing may be selective and shall be based upon all available documentation.

Dependencies:

ADV_FSP.1 TOE and security policy

AGD_USR.1 User guidance

AGD_ADM.1 Administrator guidance

ATE_FUN.1 Functional testing

Developer action elements:

ATE_IND.2.1D The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

ATE_IND.2.1C The TOE shall be suitable for testing.

Evaluator action elements:

ATE_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2E The evaluator shall test the TSF to confirm that the TSF operates as specified.

ATE_IND.2.3E **The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.**

ATE_IND.3 Independent testing - complete

Objectives

385 The objective is to demonstrate that all security functions perform as specified.

386 In this component, the objective is to repeat the developer testing.

Application notes

387 The suitability of the TOE for testing is based on the access to the TOE, and the supporting documentation and information required to run tests. The need for documentation is supported by the dependencies to other assurance families.

388 Additionally, suitability of the TOE for testing may be based on other considerations e.g., the version of the TOE submitted by the developer is not the final version.

389 The developer is required to perform testing and to provide test documentation and test results. This is addressed by the ATE_FUN family.

Dependencies:

ADV_FSP.1 TOE and security policy

AGD_USR.1 User guidance

AGD_ADM.1 Administrator guidance

ATE_FUN.1 Functional testing

Developer action elements:

ATE_IND.3.1D The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

ATE_IND.3.1C The TOE shall be suitable for testing.

Evaluator action elements:

ATE_IND.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.3.2E The evaluator shall test the TSF to confirm that the TSF operates as specified.

ATE_IND.3.3E The evaluator shall execute **all** tests in the test documentation to verify the developer test results.

Class AVA

Vulnerability assessment

390 The class “Vulnerability assessment” encompasses four families: covert channel analysis (AVA_CCA), misuse (AVA_MSU), strength of TOE security functions (AVA_SOF) and vulnerability analysis (AVA_VLA). The class addresses the existence of exploitable covert channels, the misuse or incorrect configuration of the TOE, the ability for all critical security mechanisms to withstand direct attack and the definition and assessment of penetration tests to exploit vulnerabilities introduced in the development or the operation of the TOE.

391 Figure 5.8 shows the families within this class, and the hierarchy of components within the families.

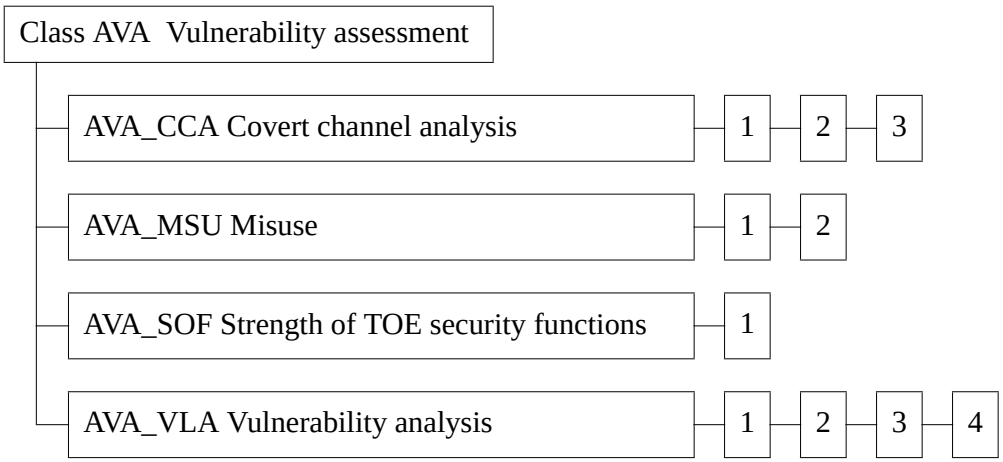

Figure 5.8 -Vulnerability assessment class decomposition

AVA_CCA Covert channel analysis

Objectives

392 Covert channel analysis is carried out to determine the existence and potential capacity of unintended signalling channels that may be exploited by malicious code.

393 The assurance requirements address the threat that unintended and exploitable signalling paths exist which may be exercised to violate the security policy.

Component levelling

394 The components are levelled on increasing rigour of covert channel analysis.

Application notes

395 Channel capacity estimations are based upon informal engineering measurements, as well as actual test measurements.

396 Details of the assumptions upon which the covert channel analysis is based shall be given, e.g., processor speed, configuration, memory, and cache size.

397 Test parameters details are (e.g., processor speed, memory and cache size), relevant configuration parameters, how the channel was exercised, used to obtain the capacity during testing.

398 The selective validation of the covert channel analysis through testing allows the evaluator the opportunity to verify any aspect of the covert channel analysis (e.g., identification, capacity estimation, elimination, monitoring, and exploitation scenarios). This does not impose a requirement to demonstrate the entire set of covert channel analysis results.

399 If there are no information flow control policies in the ST, this family of assurance requirements is no longer applicable since this family only applies to information flow control policies. Even if there are no specific functional requirements (e.g., FDP_IFF.1 to FDP_IFF.3) for eliminating, limiting, or monitoring covert channels, this family still requires the identification of covert channels.

AVA_CCA.1 Covert channel analysis

Objectives

400 The objective is to identify covert channels which are identifiable through analysis.

401 In this component, the objective is to perform informal search for covert storage channels.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_IMP.1 Subset of the implementation of the TSF

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_CCA.1.1D **The developer shall conduct a search for covert channels for each information flow control policy.**

AVA_CCA.1.2D **The developer shall provide covert channel analysis documentation.**

Content and presentation of evidence elements:

AVA_CCA.1.1C **The analysis documentation shall identify covert channels.**

AVA_CCA.1.2C **The analysis documentation shall describe the procedures used for determining the existence of covert channels, and the information needed to carry out the covert channel analysis.**

AVA_CCA.1.3C **The analysis documentation shall describe all assumptions made during the covert channel analysis.**

AVA_CCA.1.4C **The analysis documentation shall describe the method used for estimating channel capacity, which shall be based on worst case scenarios.**

AVA_CCA.1.5C **The analysis documentation shall describe the worst case exploitation scenario for each identified covert channel.**

AVA_CCA.1.6C **The analysis documentation shall provide evidence that the method used to identify covert channels is informal.**

Evaluator action elements:

AVA_CCA.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

AVA_CCA.1.2E **The evaluator shall confirm that the results of the covert channels analysis meet the functional requirements.**

AVA_CCA.1.3E **The evaluator shall selectively validate the covert channel analysis through testing.**

AVA_CCA.2 Systematic covert channel analysis

Objectives

402 The objective is to identify covert channels which are identifiable through analysis.

403 In this component, the objective is to perform a systematic search for covert channels.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_IMP.2 Implementation of the TSF

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_CCA.2.1D The developer shall conduct a search for covert channels for each information flow control policy.

AVA_CCA.2.2D The developer shall provide covert channel analysis documentation.

Content and presentation of evidence elements:

AVA_CCA.2.1C The analysis documentation shall identify covert channels.

AVA_CCA.2.2C The analysis documentation shall describe the procedures used for determining the existence of covert channels, and the information needed to carry out the covert channel analysis.

AVA_CCA.2.3C The analysis documentation shall describe all assumptions made during the covert channel analysis.

AVA_CCA.2.4C The analysis documentation shall describe the method used for estimating channel capacity, which shall be based on worst case scenarios.

AVA_CCA.2.5C The analysis documentation shall describe the worst case exploitation scenario for each identified covert channel.

AVA_CCA.2.6C The analysis documentation shall provide evidence that the method used to identify covert channels is **systematic**.

Evaluator action elements:

AVA_CCA.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_CCA.2.2E The evaluator shall confirm that the results of the covert channels analysis meet the functional requirements.

AVA_CCA.2.3E The evaluator shall selectively validate the covert channel analysis through testing.

AVA_CCA.3 Exhaustive covert channel analysis

Objectives

404 The objective is to identify covert channels which are identifiable through analysis.

405 In this component, the objective is to perform an exhaustive search for covert channels.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_IMP.2 Implementation of the TSF

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_CCA.3.1D The developer shall conduct a search for covert channels for each information flow control policy.

AVA_CCA.3.2D The developer shall provide covert channel analysis documentation.

Content and presentation of evidence elements:

AVA_CCA.3.1C The analysis documentation shall identify covert channels.

AVA_CCA.3.2C The analysis documentation shall describe the procedures used for determining the existence of covert channels, and the information needed to carry out the covert channel analysis.

AVA_CCA.3.3C The analysis documentation shall describe all assumptions made during the covert channel analysis.

AVA_CCA.3.4C The analysis documentation shall describe the method used for estimating channel capacity, which shall be based on worst case scenarios.

AVA_CCA.3.5C The analysis documentation shall describe the worst case exploitation scenario for each identified covert channel.

AVA_CCA.3.6C The analysis documentation shall provide evidence that the method used to identify covert channels is **exhaustive**.

Evaluator action elements:

AVA_CCA.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_CCA.3.2E The evaluator shall confirm that the results of the covert channels analysis meet the functional requirements.

AVA_CCA.3.3E The evaluator shall selectively validate the covert channel analysis through testing.

AVA_MSU Misuse

Objectives

406 Misuse investigates whether the TOE can be configured or used in a manner which is insecure but which an administrator or end-user of the TOE would reasonably believe to be secure.

407 The objective is to minimise the risk of human or other errors in operation which may deactivate, disable, or fail to activate security functions.

408 The objective is to minimise the probability of configuring or installing the TOE in a way which is insecure, without the end user or administrator being able to recognise it.

Component levelling

409 The components are levelled on the increasing rigour of analysis.

Application notes

410 Conflicting, misleading or incomplete guidance may result in a user of the TOE believing that the TOE is secure, when it is not. Conflicting guidance can result in vulnerabilities.

411 An example of conflicting guidance would be two guidance instructions which imply different outcomes when the same input is supplied.

412 An example of misleading guidance would be the description of a single guidance instruction which could be parsed in more than one way, one of which may result in an insecure state.

413 An example of completeness would be referencing assertions of dependencies on external security measures e.g., such as external procedural, physical and personnel controls.

AVA_MSU.1 Misuse analysis - obvious flaws

Objectives

414 The objective is to ensure that conflicting guidance in the guidance documentation have been addressed.

Dependencies:

ADO_IGS.1 Installation, generation, and start-up procedures

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_MSU.1.1D **The developer shall document an analysis of the guidance documentation for conflicting and incomplete guidance.**

AVA_MSU.1.2D **The developer shall ensure that the guidance documentation contains no misleading or unreasonable guidance.**

Content and presentation of evidence elements:

AVA_MSU.1.1C **The analysis documentation shall provide a rationale that demonstrates that the guidance is not conflicting and is complete.**

Evaluator action elements:

AVA_MSU.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

AVA_MSU.1.2E **The evaluator shall determine that there is no misleading or unreasonable guidance in the guidance documentation.**

AVA_MSU.1.3E **The evaluator shall repeat any procedures in the guidance documentation to ensure that they produce the documented results.**

AVA_MSU.2 Misuse analysis - independent verification

Objectives

415 The objective is to ensure that conflicting guidance in the guidance documentation have been addressed.

416 In this component, the objective is to provide additional assurance by performing an independent analysis.

Dependencies:

ADO_IGS.1 Installation, generation, and start-up procedures

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_MSU.2.1D The developer shall document an analysis of the guidance documentation for conflicting and incomplete guidance.

AVA_MSU.2.2D The developer shall ensure that the guidance documentation contains no misleading or unreasonable guidance.

Content and presentation of evidence elements:

AVA_MSU.2.1C The analysis documentation shall provide a rationale that demonstrates that the guidance is not conflicting and is complete.

Evaluator action elements:

AVA_MSU.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_MSU.2.2E The evaluator shall determine that there is no misleading or unreasonable guidance in the guidance documentation.

AVA_MSU.2.3E The evaluator shall repeat any procedures in the guidance documentation to ensure that they produce the documented results.

AVA_MSU.2.4E **The evaluator shall perform independent testing to confirm that the TOE can be configured and operated securely using only the guidance documentation.**

AVA_SOF Strength of TOE security functions

Objectives

417 Even if a TOE security function cannot be bypassed, deactivated, or corrupted, it may still be possible to defeat it because there is a vulnerability in the concept of its underlying security mechanisms. For those functions a qualification of their security behaviour can be made using the results of a quantitative or statistical analysis of the security behaviour of these mechanisms and the effort required to overcome them. The qualification is made in the form of a strength of TOE security functions claim.

Component levelling

418 There is only one component in this family.

Application notes

419 Security functions are implemented by security mechanisms. For example, a password mechanism can be used in the implementation of the identification and authentication security function.

420 The strength of TOE security functions evaluation is performed at the level of the security mechanism, but its results provide knowledge about the ability of the related security function to counter the identified threats.

421 The strength of a function is rated ‘basic’ if the analysis shows that the function provides adequate protection against unintended or casual breach of TOE security by attackers possessing a low attack potential.

422 The strength of a function is rated ‘medium’ if the analysis shows that the function provides adequate protection against attackers possessing a moderate attack potential.

423 The strength of a function is rated ‘high’ if the analysis shows that the function provides adequate protection against attackers possessing a high attack potential.

424 The attack potential is derived from the attacker’s expertise, opportunities, resources, and motivation.

AVA_SOF.1 Strength of TOE security function evaluation

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

Developer action elements:

AVA_SOF.1.1D **The developer shall identify all TOE security mechanisms for which a strength of TOE security function analysis is appropriate.**

AVA_SOF.1.2D **The developer shall perform a strength of TOE security function analysis for each identified mechanism.**

Content and presentation of evidence elements:

AVA_SOF.1.1C **The strength of TOE security function analysis shall determine the impact of the identified TOE security mechanisms on the ability of the TOE security functions to counter the threats.**

AVA_SOF.1.2C **The strength of TOE security function analysis shall demonstrate that the identified strength of the security functions is consistent with the security objectives of the TOE.**

AVA_SOF.1.3C **Each strength claim shall be either basic, medium, or high.**

Evaluator action elements:

AVA_SOF.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

AVA_SOF.1.2E **The evaluator shall confirm that all TOE security mechanisms requiring a strength analysis have been identified.**

AVA_SOF.1.3E **The evaluator shall confirm that the strength claims are correct.**

AVA_VLA Vulnerability analysis

Objectives

425 Vulnerability analysis is an assessment to determine whether vulnerabilities identified, during the evaluation of the construction and anticipated operation of the TOE or e.g., by flaw hypotheses, could allow malicious users to violate the TSP.

426 Vulnerability analysis deals with the threats that a malicious user will be able to discover flaws that will allow access to resources (e.g., data), allow the ability to interfere with or alter the TSF, or interfere with the authorised capabilities of other users.

Component levelling

427 Levelling is based on an increasing rigour of vulnerability analysis by the evaluator.

Application notes

428 The vulnerability analysis should consider the contents of all the TOE deliverables for the targeted evaluation assurance level.

429 Obvious vulnerabilities are those that allow common attacks or those that might be suggested by the TOE interface description. Obvious vulnerabilities are those in the public domain, details of which should be known to a developer or available from an evaluation oversight body.

430 The evidence identifies all the TOE documentation upon which the search for flaws was based.

AVA_VLA.1 Developer vulnerability analysis

Objectives

431 A vulnerability analysis is performed by the developer to ascertain the presence of “obvious” security vulnerabilities.

432 The objective is to confirm that no identified security vulnerabilities can be exploited in the intended environment for the TOE.

Application notes

433 Obvious vulnerabilities are those which are open to exploitation which requires a minimum of understanding of the TOE, skill, technical sophistication, and resources.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_VLA.1.1D **The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.**

AVA_VLA.1.2D **The developer shall document the disposition of identified vulnerabilities.**

Content and presentation of evidence elements:

AVA_VLA.1.1C **The evidence shall show, for each vulnerability, that the vulnerability cannot be exploited in the intended environment for the TOE.**

Evaluator action elements:

AVA_VLA.1.1E **The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

AVA_VLA.1.2E **The evaluator shall conduct penetration testing, based on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.**

AVA_VLA.2 Independent vulnerability analysis

Objectives

434 A vulnerability analysis is performed by the developer to ascertain the presence of "obvious" security vulnerabilities.

435 The objective is to confirm that no identified security vulnerabilities can be exploited in the intended environment for the TOE.

436 An independent vulnerability analysis is performed by the evaluator, which goes beyond the "obvious" security vulnerabilities. The analysis considers the deliverables available for the targeted evaluation assurance level.

Application notes

437 Obvious vulnerabilities are those which are open to exploitation which requires a minimum of understanding of the TOE, skill, technical sophistication, and resources.

438 Independent vulnerability analysis is based on fairly detailed technical information. The attacker is assumed to be only reasonably familiar with the specific implementation of the TOE. The attacker is presumed to have a reasonable level of technical sophistication.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ADV_IMP.1 Subset of the implementation of the TSF

ADV_LLD.1 Descriptive low-level design

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_VLA.2.1D The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.

AVA_VLA.2.2D The developer shall document the disposition of identified vulnerabilities.

Content and presentation of evidence elements:

AVA_VLA.2.1C The evidence shall show, for each vulnerability, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.2.2C **The documentation shall justify that the TOE, with the identified vulnerabilities, is resistant to obvious penetration attacks.**

Evaluator action elements:

AVA_VLA.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.2.2E The evaluator shall conduct penetration testing, based on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

AVA_VLA.2.3E **The evaluator shall perform an independent vulnerability analysis.**

AVA_VLA.2.4E **The evaluator shall perform independent penetration testing, based on the independent vulnerability analysis, to determine the exploitability of identified vulnerabilities in the target environment.**

AVA_VLA.2.5E **The evaluator shall determine that the TOE is resistant to obvious penetration attacks.**

AVA_VLA.3 Relatively resistant

Objectives

439 A vulnerability analysis is performed by the developer to ascertain the presence of “obvious” security vulnerabilities.

440 The objective is to confirm that no identified security vulnerabilities can be exploited in the intended environment for the TOE.

441 An independent vulnerability analysis is performed by the evaluator, which goes beyond the “obvious” security vulnerabilities. The analysis considers the deliverables available for the targeted evaluation assurance level.

442 In addition, the independent vulnerability analysis performed by the evaluator is based on analytical techniques which are employed to discover vulnerabilities that would require sophisticated attackers.

443 The TOE must be shown to be relatively resistant to penetration attack.

Application notes

444 Obvious vulnerabilities are those which are open to exploitation which requires a minimum of understanding of the TOE, skill, technical sophistication, and resources.

445 Independent vulnerability analysis is based on detailed technical information. The attacker is assumed to be thoroughly familiar with the specific implementation of the TOE. The attacker is presumed to have a moderate level of technical sophistication.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ADV_IMP.1 Subset of the implementation of the TSF

ADV_LLD.1 Descriptive low-level design

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_VLA.3.1D The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.

AVA_VLA.3.2D The developer shall document the disposition of identified vulnerabilities.

Content and presentation of evidence elements:

AVA_VLA.3.1C The evidence shall show, for each vulnerability, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.3.2C The documentation shall justify that the TOE, with the identified vulnerabilities, is **relatively** resistant to **penetration attacks**.

Evaluator action elements:

AVA_VLA.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.3.2E The evaluator shall conduct penetration testing, based on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

AVA_VLA.3.3E The evaluator shall perform an independent vulnerability analysis.

AVA_VLA.3.4E The evaluator shall perform independent penetration testing, based on the independent vulnerability analysis, to determine the exploitability of identified vulnerabilities in the target environment.

AVA_VLA.3.5E The evaluator shall determine that the TOE is **relatively** resistant to penetration attacks.

AVA_VLA.4 Highly resistant

Objectives

446 A vulnerability analysis is performed by the developer to ascertain the presence of "obvious" security vulnerabilities.

447 The objective is to confirm that no identified security vulnerabilities can be exploited in the intended environment for the TOE.

448 An independent vulnerability analysis is performed by the evaluator, which goes beyond the "obvious" security vulnerabilities. The analysis considers the deliverables available for the targeted evaluation assurance level.

449 In addition, the independent vulnerability analysis performed by the evaluator is based on analytical techniques which are employed to discover vulnerabilities that would require sophisticated attackers.

450 The TOE must be shown to be highly resistant to penetration attacks.

Application notes

451 Obvious vulnerabilities are those which are open to exploitation which requires a minimum of understanding of the TOE, skill, technical sophistication, and resources.

452 Independent vulnerability analysis is based on highly detailed technical information. The attacker is assumed to be thoroughly familiar with the specific implementation of the TOE. The attacker is presumed to have a high level of technical sophistication.

Dependencies:

ADV_FSP.1 TOE and security policy

ADV_HLD.1 Descriptive high-level design

ADV_IMP.1 Subset of the implementation of the TSF

ADV_LLD.1 Descriptive low-level design

AGD_ADM.1 Administrator guidance

AGD_USR.1 User guidance

Developer action elements:

AVA_VLA.4.1D The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.

AVA_VLA.4.2D The developer shall document the disposition of identified vulnerabilities.

Content and presentation of evidence elements:

AVA_VLA.4.1C The evidence shall show, for each vulnerability, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.4.2C The documentation shall justify that the TOE, with the identified vulnerabilities, is **highly** resistant to penetration attacks.

AVA_VLA.4.3C **The analysis documentation shall provide a justification that the analysis completely addresses the TOE deliverables.**

Evaluator action elements:

AVA_VLA.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.4.2E The evaluator shall conduct penetration testing, based on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

AVA_VLA.4.3E The evaluator shall perform an independent vulnerability analysis.

AVA_VLA.4.4E The evaluator shall perform independent penetration testing, based on the independent vulnerability analysis, to determine the exploitability of identified vulnerabilities in the target environment.

AVA_VLA.4.5E The evaluator shall determine that the TOE is **highly** resistant to penetration attacks.

Annex A

Cross reference of assurance component dependencies

453 The dependencies documented in the components in Chapter 5 are the direct dependencies between the assurance components. Table A.1 summarises both the direct dependencies and the indirect dependencies. The indirect dependencies are the cumulative result of iteratively including all the dependencies of each component identified as being a dependency.

Comp. Names	TSS	AUT	CAP	SCP	DEL	IGS	FSP	HLD	IMP	INT	LLD	RCR	ADM	USR	DVS	FLR	LCD	TAT	COV	DPT	FUN	IND	CCA	MSU	SOF	VLA
AUT.X			**2**	*1*											*1*											
CAP.1																										
CAP.2-3			*2*	**1**											**1**											
CAP.4			*2*	**1**											**2**											
SCP.X			**2**	*1*											*1*											
DEL.1																										
DEL.2-3			**2**	*1*											*1*											
IGS.X	*1*						*1*					*1*	**1**													
FSP.1-6	**1**											**1**														
HLD.1-2	*1*						**1**					**1**														
HLD.3-4	*1*						**3**					**2**														
HLD.5	*1*						**4**					**3**														
IMP.1	*1*						*1*	*2*			**1**	**1**						**1**								
IMP.2	*1*						*1*	*2*	*1*		**1**	**1**						**2**								
IMP.3	*1*						*1*	*2*	*1*	**1**	**1**	**1**						**3**								
INT.1-2	*1*						*1*	*2*	**1**		**1**	*1*						*1*								
INT.3	*1*						*1*	*2*	**2**		**1**	*1*						*2*								
LLD.1	*1*						*1*	**1**				**1**														
LLD.2	*1*						*3*	**3**				**2**														
LLD.3	*1*						*3*	**5**				**3**														
RCR.X																										
ADM.1	*1*						**1**					*1*														
USR.1	*1*						**1**					*1*														
DVS.X																										
FLR.1																										

Table A.1 -Assurance component dependencies[a]

Comp. Names	TSS	AUT	CAP	SCP	DEL	IGS	FSP	HLD	IMP	INT	LLD	RCR	ADM	USR	DVS	FLR	LCD	TAT	COV	DPT	FUN	IND	CCA	MSU	SOF	VLA
FLR.2-4	*1*						*1*					*1*	**1**													
LCD.X																										
TAT.1																										
TAT.2-3	*1*						*1*	*2*	**1**		*1*	*1*						*1*								
COV.1-3	*1*						**1**					*1*							*1*	*1*	**1**					
DPT.1	*1*						**1**					*1*							*1*	*1*	**1**					
DPT.2	*1*						**1**	**1**				*1*							*1*	*1*	**1**					
DPT.3	*1*						**1**	**1**			**1**	*1*							*1*	*1*	**1**					
DPT.4	*1*						**1**	**1**	**2**		**1**	*1*						*2*	*1*	*1*	**1**					
FUN.1	*1*						*1*					*1*							**1**	**1**	*1*					
IND.1	*1*						**1**					*1*	**1**	**1**												
IND.2-3	*1*						**1**					*1*	**1**	**1**					*1*	*1*	**1**					
CCA.1	*1*						**1**	*2*	**1**		*1*	*1*	**1**	**1**				*1*								
CCA.2-3	*1*						**1**	*2*	**2**		*1*	*1*	**1**	**1**				*2*								
MSU.X	*1*					**1**	*1*					*1*	**1**	**1**												
SOF.1	*1*						**1**	**1**				*1*														
VLA.1	*1*						**1**	**1**				*1*	**1**	**1**												
VLA.2-4	*1*						**1**	**1**	**1**		**1**	*1*	**1**	**1**				*1*								

Table A.1 -Assurance component dependencies[a]

a. In Table A.1, the left column represents groupings of specific components (using only the last three digits of the component name and an indicator of component number, range of numbers, or “X” for all numbers). Each non-empty box in the table indicates a specific component, identified by a its name at the top of the column and the number in the box, on which the component in the left column is dependent. Bold numbers represent direct dependencies. Italicised numbers represent indirect dependencies. Dark shading represents the intersection of a component with itself.

Annex B

Cross reference of EALs and assurance components

454 Table B.1 describes the relationship between the evaluation assurance levels and the assurance classes, families and components.

Assurance Class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Configuration management	ACM_AUT				**1**	1	**2**	2
	ACM_CAP	**1**	1	**2**	**3**	3	**4**	4
	ACM_SCP			**1**	**2**	**3**	3	3
Delivery and operation	ADO_DEL							
	ADO_IGS		**1**	1	1	1	1	1
Development	ADV_FSP	**1**	1	1	**2**	**4**	**5**	**6**
	ADV_HLD		**1**	**2**	2	**3**	**4**	**5**
	ADV_IMP				**1**	**2**	**3**	3
	ADV_INT					**1**	**2**	**3**
	ADV_LLD				**1**	1	**2**	2
	ADV_RCR	**1**	1	1	1	**2**	2	**3**
Guidance documents	AGD_ADM	**1**	1	1	1	1	1	1
	AGD_USR	**1**	1	1	1	1	1	1
Life cycle support	ALC_DVS			**1**	1	1	**2**	2
	ALC_FLR							
	ALC_LCD				**1**	**2**	2	**3**
	ALC_TAT				**1**	**2**	**3**	3
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT		**1**	**2**	2	**3**	3	**4**
	ATE_FUN		**1**	1	1	1	1	1
	ATE_IND	**1**	1	**2**	2	2	2	**3**
Vulnerability assessment	AVA_CCA					**1**	**2**	2
	AVA_MSU			**1**	**2**	2	2	2
	AVA_SOF		**1**	1	1	1	1	1
	AVA_VLA		**1**	1	**2**	**3**	**4**	4

Table B.1 -Evaluation Assurance Level Summary

Annex C

CC observation report (CCOR)

C.1 Introduction

455 The CC sponsoring organisations welcome feedback from the community and are particularly interested in observations and comments arising out of trial application of the criteria.

456 The CC sponsoring organisations have set up a body, the Common Criteria Implementation Board (CCIB), to coordinate and learn from the community experience and to ensure that future issues of the CC can benefit from that experience.

457 Comments, observations, and requests for interpretations should be sent to one of the addresses listed inside the front cover of the CC. If you require feedback on a specific evaluation matter, you should use the contact address which corresponds to the evaluation authority concerned.

C.2 Categorisation of observation report

458 In order to allow automated categorisation of the observations, a standard observation format is needed. Each observation should include an identifier as to whether the comment pertains to the ***approach*** in the CC, the technical ***detail*** of any specific portion of the CC, or ***editorial*** work that needs to be done. Additionally, for comments on technical detail, an indication of the scope of the comment (e.g., ***local***, ***global***) should be provided.

459 The following provides a description of each of these terms:

a) *Approach*: observations requesting further guidance relating to the approach of the CC which the author of the observation report considers to be fundamental to the further progress of the CC or trial application of the criteria should be marked with this identifier.

b) *Detail*: Specific observations on technical details of the CC should be marked with this identifier. These comments should be further categorised as either local or global.

Local: is applicable to a single specific class, family, component, or element.

Global: is applicable to multiple classes, families, components, or elements.

c) *Editorial*: typographical and grammatical errors, as well as comments on presentation style.

Local: is applicable to a single specific class, family, component, or element.

Global: is applicable to multiple classes, families, components, or elements.

C.3 Format of observation report

460 The following provides a description of each of the structure of the required comment format and an example of a comment in the required format.

461 If you are submitting one or more observations by electronic mail or other machine readable format, please insert the tags defined below starting in the first column as this will greatly assist in any automated handling of your input.

462 Each observation report should consist of three parts.

a) The first part consists of a tags **$1:** to **$4:**, which includes the information to allow the unique identification of the originator. This first set of tags is required only once per single observation or batch of observations.

b) The second part consists of tags **$5**: to **$9:**, which includes the information to allow the unique identification and categorisation of the observation, the actual observation itself and suggested solution. The text of each observation should extend to as many lines as are needed to fully express the observation. There can be one or more observations in an observation report.

The set of tags **$5**: to **$9:**, comprising this second part of the observation report, should be repeated for each observation being submitted.

c) The third part consists of a single terminating tag **$$:**. This final tag is required only once per single observation or batch of observations.

C.3.1 Tag definitions for observation report

$1: Originator name

463 Name of commenter (only required once per message).

$2: Originator organisation

464 Originator organisation/affiliation (only required once per message).

$3: Return address

465 Electronic mail or other address for response (only required once per message).

$4: Date

466 Submission date of observation YY/MM/DD (only required once per message).

$5: Originator report reference identification

467 Reference for observation which is unique to originator. Please include your initials or similar unique discriminator, e.g., ABC1234.

$6: One line summary/title of observation

468 Short summary/title for problem (up to 60 characters).

$7: CC document reference

469 Single reference to the affected area of the CC as detailed as appropriate. Where possible, part number, section, paragraph, class, family, component, or requirement reference should be provided.

470 The template for CC document reference is as follows:

$7: Part / Section / Paragraph / [Approach / Detail - [Local / Global] / Editorial] - [Local / Global] / [Keyword]

471 The CC document reference template should be completed as follows (see below for completed example):

a) The characters "**$7:**", to indicate the start of an observation.

b) Identification of the CC part, section, and paragraph to which the comment applies in the CC. All 3 pieces of identifying information should be provided, each separated by a slash character (/).

Valid identifiers for the CC Part are e.g., part 1 or 1, part 2 or 2, part 3 or 3, and profiles or PP.

Identification for the CC section should be either a section number (e.g., 1.3.2), if applicable, or, for requirement classes, families, or components, the name of the class (e.g., FIA), family (e.g., FIA_ATD), or component (e.g., FIA_ATD.1).

c) Identification of the reviewer's categorisation of the observation. Brackets "[..]" indicate that the reviewer should choose *one* of the options contained within the brackets, these can be abbreviated to the initial character only (e.g., "A" , "D - L", or "E - G").

d) An optional keyword.

472 Any identification field should be left blank or be filled with an asterisk (*) to indicate that the field is not applicable or necessary for the comment.

$8: Statement of observation

473 Comprehensive statement of observation or query, contains the actual text of the observation. Should include specific reference to examples of the observation, where appropriate.

$9: Suggested solution

474 Proposed solution or solution approach.

$$: Terminating tag.

475 This enables any automated handling to determine the end of the batch of observations (only required once per batch of observations).

C.3.2 Example observations:

$1: A. N. Other

$2: PPs ‘R’ US

$3: another@ppsrus.com

$4: 960131

$5: ano.comment.1

$6: Presentation comment.

$7: 1 / 8.1 / 90 / Editorial - Local /

$8: The word “global” at the end of the first line should be italicised.

$9: Italicise “global”.

$5: ano.comment.2

$6: Missing requirement for audit.

$7: 2 / FAU / 336 / Detail - Local /

$8: The first sentence of this paragraph is incomplete.

$9: The first sentence should include “imminent” violations.

$5: ano.comment.3

$6: Problems in navigating the document.

$7: 2 / * / * / Approach / threats

$8: The statements of threat in the functional families are largely re-statements of the family behaviour from the threat viewpoint. Does this material need to be re-stated twice within the functional families?

$9: Could all threat information be described in a separate section with a table mapping the various functional components to the threats they address?

$$: This is the end tag, the contents are immaterial.

C.4 Printed observation report

476 An example of a printed observation report is provided in Table C.1.

COMMON CRITERIA OBSERVATION REPORT	
$1:	**Originator Name**
$2:	**Originator organisation**
$3:	**Return address**
$4:	**Date**
$5:	**Originator report reference identification**
$6:	**One line summary/title of observation**
$7:	**CC document reference**
$8:	**Statement of observation**
$9:	**Suggested solution**
$$:	

Table C.1 - CC observation report